Finanziato dall'Unione europea NextGenerationEU

Ministero dell'Università e della Ricerca

Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA

Università degli Studi di Padova

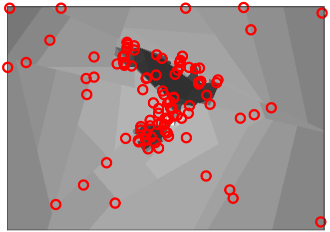
Computational harmonic analysis on graphs

2. Graph Wedgelets: geometric wavelets on graphs

Wolfgang Erb

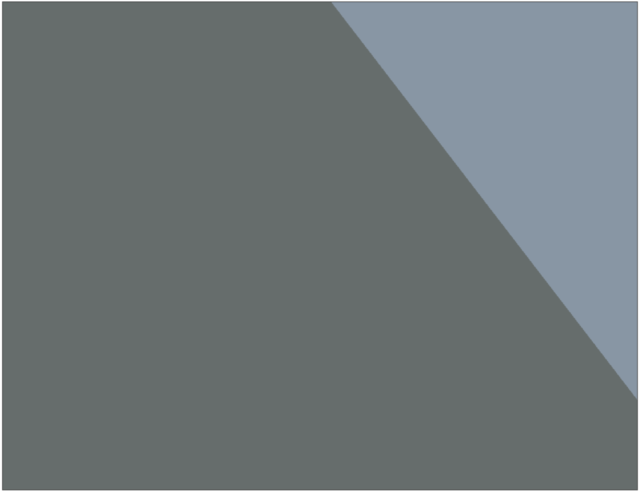
University of Padova

Workshop and Summer School on Applied Analysis 2024
September 16-20, 2024, Chemnitz, Germany



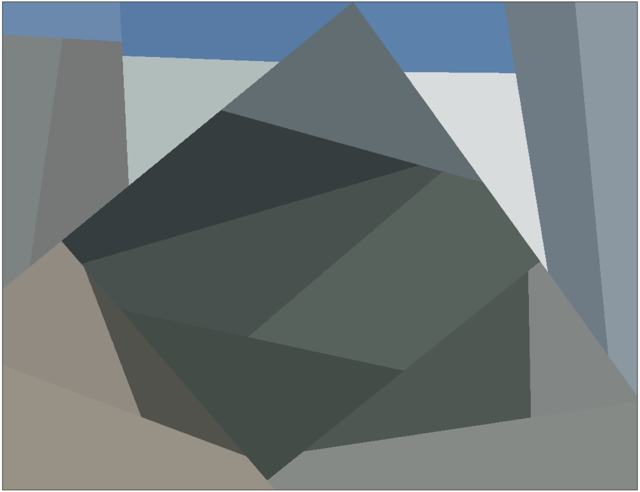
wolfgang.erb@unipd.it

What is in this picture?



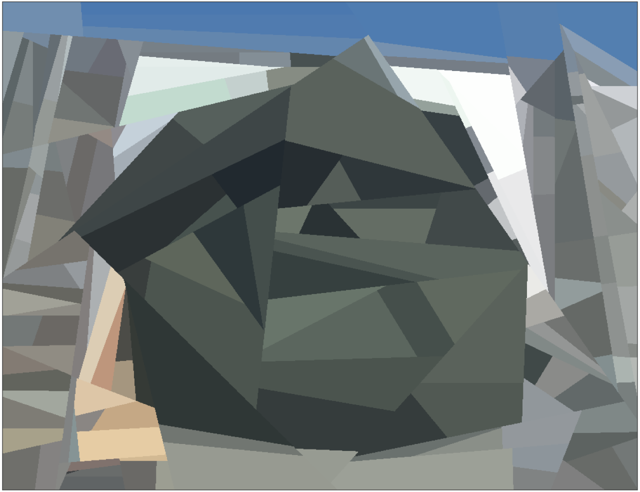
Signature image: piecewise constant approximation using 2 wedges.

What is in this picture?



This is a piecewise constant approximation of the picture using 20 wedges.

What is in this picture?



Piecewise constant approximation of the picture using 200 wedges.

What is in this picture?

Piecewise constant approximation of the picture using 2000 wedges.

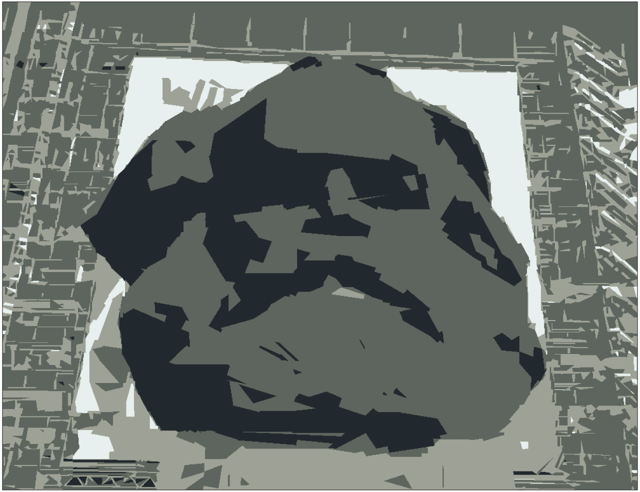
What is in this picture?

Piecewise constant approximation with 20000 wedges.

What is in this picture?

This is the original.

What is in this picture?



This is a segmentation of the image.

Goal of this presentation

Study piecewise constant approximation of graph signals (or images) by discrete wedgelets. More precisely, we will consider geometric wavelets based on binary wedge partitioning trees (BWPs) on graphs.

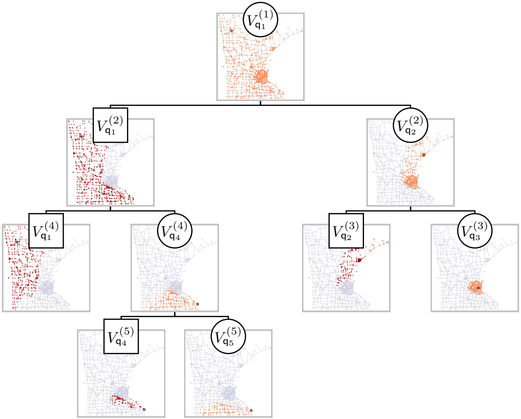


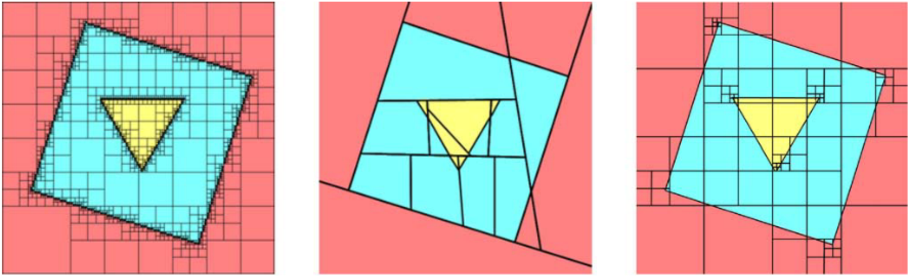
Figure 1: Binary wedge partitioning tree on the Minnesota graph

Adaptive piecewise constant approximation of images

For image decomposition and compression with piecewise constant functions there exists a large amount of literature. Most techniques are based on continuous models including a tree-based hierarchical decomposition of the image and a proper discretization.

- Classical Haar functions and Haar wavelets
- Adaptive triangulation (Cohen, Dyn, Hecht, Mirebeau, Demaret, Iske)
- Quadtrees (Leonardi, Kunt, Samet)
- Tetralets (Krommweh)
- Wedgelets (Donoho, Demaret, Friedrich, Führ, Wicker, Romberg, Wakin, Choi, Baraniuk)
- Binary space partitionings (Radha, Leonardi, Naylor, Vetterli).

Adaptive piecewise constant approximation of images

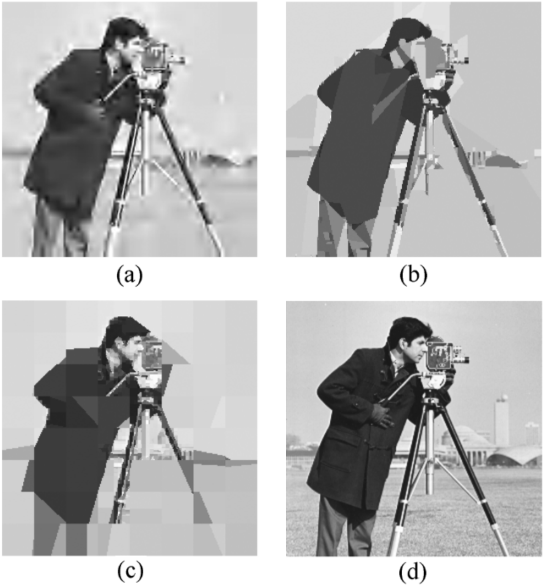


Comparison between different adaptive tree-based dictionaries for the approximation of a piecewise constant image.

Left: Quadtree
Middle: Binary space partitioning (BSP)
Right: Wedgelets (Donoho)

Images taken from Kassim, Lee, Zonoobi, IEEE Trans. Image. Process. 2009

Adaptive piecewise constant approximation of images



Comparison between different adaptive methods for the approximation of the Cameraman with 0.15bpp.

(a) JPEG2000, 16.60 dB
(b) BSP, 17.9 dB
(c) Wedgelets, 15.8 dB
(d) Original image

Images taken from Kassim, Lee, Zonoobi, IEEE Trans. Image. Process. 2009

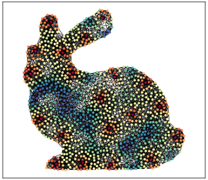
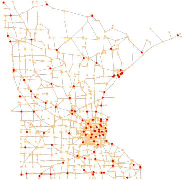
Adaptive piecewise constant approximation of images

Observation: there is an inherent trade-off in adaptive decompositions.

- High adaptivity & large dictionaries lead to sparse representations of images, the computational cost and memory expenses for the single atoms are large (for instance in binary space partitionings)
- Low adaptivity & small dictionaries require many elements to represent the image, the computational cost and memory expenses of the single atoms are low (for instance in quadtrees)

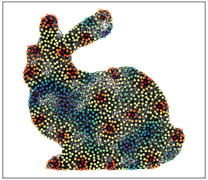
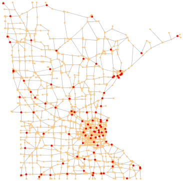
Observation in literature: Highly adaptive methods as binary space partitionings are competitive to JPEG2000 mainly in low-bit compression.

Why do we want to transfer such concepts on graphs?


- We can use graphs to describe images in a discrete way. In implementations we don't have to think about possible discretizations.
- On graphs we have efficient algorithms to calculate distances, partitions and splittings.
- Graphs are dimensionless. Once we have a particular tool for graphs, we can also use it for higher dimensional data, as for instance videos.
- The graph Laplacian offers a discrete way to measure smoothness of images/signals.

Why do we want to transfer such concepts on graphs?

Independently, graphs are interesting objects as they allow to model complex irregular structures with a simple discrete structure.

Examples:

- Social networks: nodes = persons, edges = relations
- Transport networks: nodes = crossing, edges = streets
- Images: nodes = pixels, edges connect nearby pixels.

Information on graphs is given in terms of graph signals. Also for graph signals compression techniques are needed if n gets large.

Graphs and graph signals

We consider simple and undirected graphs G given as

$$G = (V, E, \mathbf{A}),$$

i.e., with vertices

$$V = \{\mathrm{v}_1, \ldots, \mathrm{v}_n\}$$

undirected edges $E \subset V \times V$, and a symmetric adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ with non-negative entries

$$\begin{cases} \mathbf{A}_{i,j} > 0 \text{ if } (\mathrm{v}_i, \mathrm{v}_j) \in E, \\ \mathbf{A}_{i,j} = 0 \text{ otherwise.} \end{cases}$$

Further, we need a metric distance d between the nodes.

Standard $\mathbf{A}$: only entries 1 (if there is an edge) and 0.

Standard d: the length of the shortest path connecting two nodes.

Graphs and graph signals

Graph signals are mappings $f : V \to \mathbb{R}$ (or $f : V \to \mathbb{C}$)

- A signal f is defined on the vertices $\mathrm{v} \in V$ of the graph
- For a graph with n vertices, we can represent f as a vector

$$f = [f(\mathrm{v}_1)\ f(\mathrm{v}_1)\ \cdots\ f(\mathrm{v}_n)]^* \in \mathbb{R}^n \quad (\in \mathbb{C}^n).$$

- We use $\mathcal{L}(V)$ to denote the linear space of all graph signals.

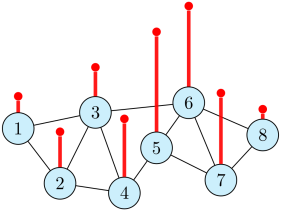


Fig.: Illustration of a graph signal f.

Wavelets on graphs for piecewise constant approximation

As in a continuous setting, wavelets on graphs can be used to

- analyse the smoothness of signals in proper spaces;
- provide a multiresolution analysis of graph signals;
- denoise signals (in terms of properly defined filters);
- compress signals on huge graphs.

Large number of available literature also on graphs

- Diffusion wavelets (Coifman, Maggioni)
- Wavelets and vertex-frequency analysis via spectral graph theory (Gribonval, Hammond, Ricaud, Shuman, Vandergheynst)
- Wavelet packets on graphs (Bulai, Saliani)

Interesting for us are wavelets for piecewise constant approximation of graph signals. One important approach is based on hierarchical partitioning trees (Gavish, Nadler, Coifman, Irion, Saito, Murtagh).

Make things adaptive - binary wedge splits

Definition 1

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

$$V' = \{\mathrm{v} \in V \mid \mathrm{d}(\mathrm{v}, \mathrm{v}') \leq \mathrm{d}(\mathrm{v}, \mathrm{v}'')\}, \quad \text{and}$$
$$V'' = \{\mathrm{v} \in V \mid \mathrm{d}(\mathrm{v}, \mathrm{v}') > \mathrm{d}(\mathrm{v}, \mathrm{v}'')\}.$$

Fig.: Illustration of a binary wedge split based on two nodes q_1 and q_2.

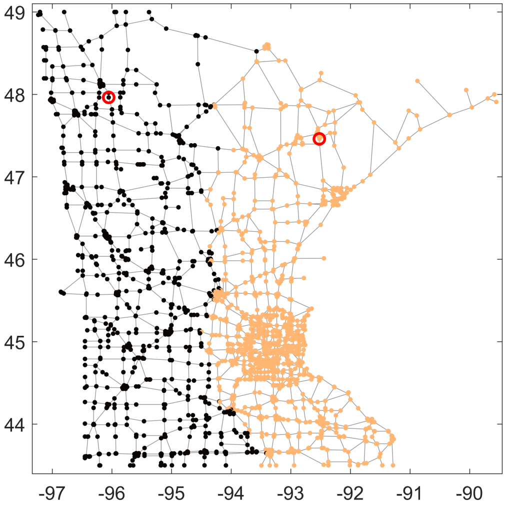


Fig.: Illustration of a binary wedge partitioning based on two nodes q_1 and q_2 (in red).

Definition 2 (Binary wedge partitioning (BWP) trees)

A binary wedge partitioning (BWP) tree $\mathcal{T}_Q$ of G w.r.t. the ordered set $Q = \{\mathrm{q}_1, \ldots, \mathrm{q}_M\} \subset V$ is a binary partitioning tree constructed as follows:

1. The root of $\mathcal{T}_Q$ is the set V. It forms the trivial partition $\mathcal{P}^{(1)} = \{V_{\mathrm{q}_1}^{(1)}\} = \{V\}$ and is associated to $\mathrm{q}_1 \in Q$.
2. For a partition $\mathcal{P}^{(m)} = \{V_{\mathrm{q}_1}^{(m)}, \ldots, V_{\mathrm{q}_m}^{(m)}\}$ of V in $\mathcal{T}_Q$ consider the node $\mathrm{q}_{m+1} \in V_{\mathrm{q}_j}^{(m)}$ for a $j \in \{1, \ldots, m\}$. We split $V_{\mathrm{q}_j}^{(m)}$ by a wedge split based on q_j and q_{m+1} into two disjoint sets $V_{(\mathrm{q}_j,\mathrm{q}_{m+1})}^{(m)+}$ and $V_{(\mathrm{q}_j,\mathrm{q}_{m+1})}^{(m)-}$ and obtain the new partition

$$\mathcal{P}^{(m+1)} = \{V_{\mathrm{q}_1}^{(m+1)}, \ldots, V_{\mathrm{q}_{m+1}}^{(m+1)}\}$$

with $V_{\mathrm{q}_i}^{(m+1)} = V_{\mathrm{q}_i}^{(m)}$ if $i \neq \{j, m+1\}$, $V_{\mathrm{q}_j}^{(m+1)} = V_{(\mathrm{q}_j,\mathrm{q}_{m+1})}^{(m)+}$ and $V_{\mathrm{q}_{m+1}}^{(m+1)} = V_{(\mathrm{q}_j,\mathrm{q}_{m+1})}^{(m)-}$.

Binary wedge partitioning trees

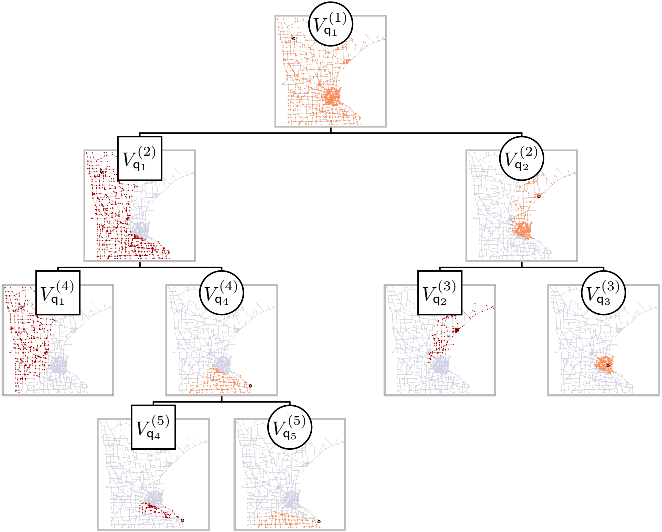


Fig.: Again the illustration of the BWP tree on the Minessota graph.

Advantage: the tree depends only on the ordered set Q.

Proposition 3

Let $\mathcal{T}_Q$ be a BWP tree determined by the ordered set $Q = \{\mathrm{q}_1, \ldots, \mathrm{q}_M\}$.

1. *A BWP tree $\mathcal{T}_Q$ contains $2M-1$ elements: 1 root, $2M-2$ children.*
2. *The M leaves of the binary tree $\mathcal{T}_Q$ are given by the elements of the M-th. partition $\mathcal{P}^{(M)} = \{V_{\mathrm{q}_1}^{(M)}, \ldots, V_{\mathrm{q}_M}^{(M)}\}$.*
3. *A BWP tree $\mathcal{T}_Q$ is complete if and only if $|Q| = |V| = n$.*
4. *A BWP tree $\mathcal{T}_Q$ is balanced with $\frac{1}{2} \leq \rho \leq \frac{n-1}{n}$.*

Definition 4

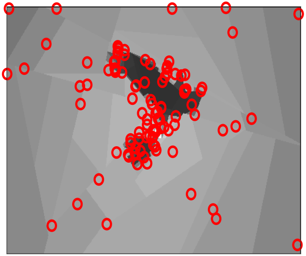
The characteristic functions

$$\omega_{\mathrm{q}_i}^{(m)}(\mathrm{v}) = \chi_{V_{\mathrm{q}_i}^{(m)}}(\mathrm{v}), \quad 1 \leq i \leq m,\ 1 \leq m \leq M,$$

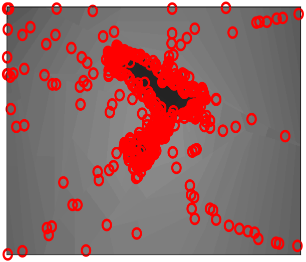
of the sets $V_{\mathrm{q}_i}^{(m)}$ will be referred to as wedgelets with respect to the BWP tree $\mathcal{T}_Q$. The wedgelets $\{\omega_{\mathrm{q}_i}^{(m)} : 1 \leq i \leq m\}$ form an orthogonal basis for the piecewise constant functions on the partition $\mathcal{P}^{(m)}$.

Question: how do we get adaptive wedge splits to approximate functions?

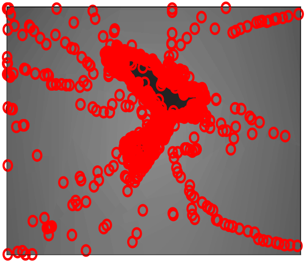
Original image



100 wedgelets



500 wedgelets



1000 wedgelets

Greedy generation of BWP trees I

There are several possibilities to generate BWP trees

1) Max-distance (MD) greedy wedge splitting: at stage m, the domain $V_{\mathrm{q}_j}^{(m)}$ with the maximal $\mathcal{L}^2$-error is chosen, i.e.

$$j = \underset{i \in \{1,\ldots,m\}}{\operatorname{argmax}} \|f - \bar{f}_{V_{\mathrm{q}_i}^{(m)}}\|_{\mathcal{L}^2(V_{\mathrm{q}_i}^{(m)})}, \tag{1}$$

where

$$\bar{f}_{V_{\mathrm{q}_i}^{(m)}} = \frac{\langle f, \omega_{\mathrm{q}_i}^{(m)} \rangle}{|V_{\mathrm{q}_i}^{(m)}|} = \frac{1}{|V_{\mathrm{q}_i}^{(m)}|} \sum_{\mathrm{v} \in V_{\mathrm{q}_i}^{(m)}} f(\mathrm{v}).$$

As soon as j is determined, the subsequent node set q_{m+1} is chosen by the selection rule

$$\mathrm{q}_{m+1} = \arg\max_{\mathrm{v} \in V_{\mathrm{q}_j}^{(m)}} \mathrm{d}(\mathrm{q}_j, \mathrm{v}),$$

i.e., q_{m+1} is the vertex in $V_{\mathrm{q}_j}^{(m)}$ furthest away from q_j.

Greedy generation of BWP trees II

2) Fully-adaptive (FA) greedy wedge splitting:

The subset $V_{\mathrm{q}_j}^{(m)}$ to be split is selected according to (1), i.e.,

$$j = \underset{i \in \{1,\ldots,m\}}{\operatorname{argmax}} \|f - \bar{f}_{V_{\mathrm{q}_i}^{(m)}}\|_{\mathcal{L}^2(V_{\mathrm{q}_i}^{(m)})},$$

and also the node q_{m+1} is chosen according to an adaptive rule. If $\{V_{(\mathrm{q}_j,\mathrm{q})}^{(m)+}, V_{(\mathrm{q}_j,\mathrm{q})}^{(m)-}\}$ denotes the partition of $V_{\mathrm{q}_j}^{(m)}$ for the wedge split determined by q_j and a second node q, we choose q_{m+1} such that

$$\|f - \bar{f}_{V_{(\mathrm{q}_j,\mathrm{q})}^{(m)+}}\|^2_{\mathcal{L}^2(V_{(\mathrm{q}_j,\mathrm{q})}^{(m)+})} + \|f - \bar{f}_{V_{(\mathrm{q}_j,\mathrm{q})}^{(m)-}}\|^2_{\mathcal{L}^2(V_{(\mathrm{q}_j,\mathrm{q})}^{(m)-})} \tag{2}$$

is minimized over all $\mathrm{q} \in V_{\mathrm{q}_j}^{(m)}$.

Greedy generation of BWP trees III

3) Randomized (R) greedy wedge splitting:

If the size of the subset $V_{\mathrm{q}_j}^{(m)}$ is large, an alternative to the fully-adaptive procedure is a randomized splitting strategy.

In this strategy, the minimization of the quantity (2) is performed on a subset of $1 \leq R \leq |V_{\mathrm{q}_j}^{(m)}|$ randomly picked nodes of $V_{\mathrm{q}_j}^{(m)}$.

The parameter R acts as a control parameter giving a result close or identical to FA-greedy if R is chosen large enough.

Algorithm 1: Wedgelet encoding of a graph signal

Input: Function f, first node $\mathrm{q}_1 \in V$, $\mathcal{P}^{(1)} = \{V\} = \{V_{\mathrm{q}_1}^{(1)}\}$ and size M.

for $m = 2$ *to* M **do**

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

$$j = \underset{i \in \{1,\ldots,m-1\}}{\arg\max} \left\| f - \bar{f}_{V_{\mathrm{q}_i}^{(m-1)}} \right\|_{\mathcal{L}^2(V_{\mathrm{q}_i}^{(m-1)})};$$

2) Conduct one of the following alternatives:
- **Max-distance (MD) greedy procedure**;
- **Fully-adaptive (FA) greedy procedure**;
- **Randomized (R) greedy procedure**;

3) Generate **new partition** $\mathcal{P}^{(m)}$ from the partition $\mathcal{P}^{(m-1)}$ by a wedge split of the subset $V_{\mathrm{q}_j}^{(m-1)}$ into the children sets $V_{(\mathrm{q}_j,\mathrm{q}_m)}^{(m-1)+}$ and $V_{(\mathrm{q}_j,\mathrm{q}_m)}^{(m-1)-}$;

4) Compute **mean values** $\bar{f}_{V_{\mathrm{q}_i}^{(m)}}$, $i \in \{1,\ldots,m\}$, for the new partition $\mathcal{P}^{(m)}$ by an update from $\mathcal{P}^{(m-1)}$.

Output: $Q = \{\mathrm{q}_1,\ldots,\mathrm{q}_M\}$, $\left\{\bar{f}_{V_{\mathrm{q}_1}^{(M)}},\ldots,\bar{f}_{V_{\mathrm{q}_M}^{(M)}}\right\}$.

Algorithm 2: Wedgelet decoding of a graph signal

Input: $Q = \{\mathrm{q}_1, \ldots, \mathrm{q}_M\}$, $\{\bar{f}_{V_{\mathrm{q}_1}^{(M)}}, \ldots, \bar{f}_{V_{\mathrm{q}_M}^{(M)}}\}$.

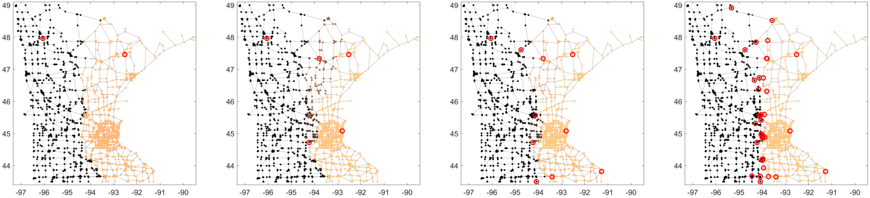
Calculate the partition $\mathcal{P}^{(M)} = \{V_{\mathrm{q}_1}^{(M)}, \ldots, V_{\mathrm{q}_M}^{(M)}\}$ of V by elementary wedge splits along the BWP tree $\mathcal{T}_Q$.

Output: The wedgelet approximation

$$\mathcal{W}_M f(\mathrm{v}) = \sum_{i=1}^{M} \bar{f}_{V_{\mathrm{q}_i}^{(M)}} \, \omega_{\mathrm{q}_i}^{(M)}(\mathrm{v})$$

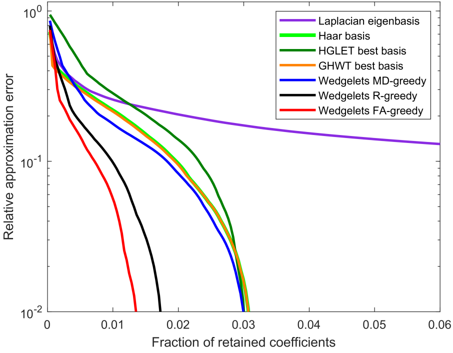
of f. For $M = n$, $\mathcal{W}_n f = f$ is reconstructed.

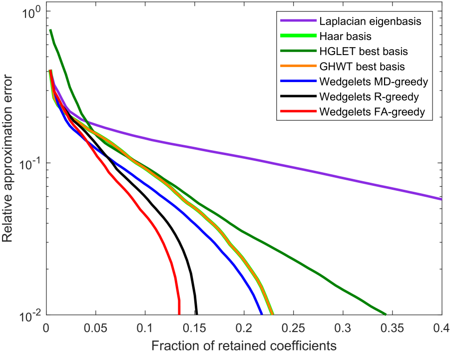
Graph signal approximation with wedgelets



Approximation of a binary function on the Minnesota graph with $1, 4, 9$ and 39 wedge splits (from left to right). The red rings indicate the center nodes Q. The number of wrongly classified nodes equals 356, 286, 110, and 12, respectively.

Comparison to non-adaptive wavelet approaches on graphs





Comparison between best *m*-term approximation using BWP wavelets (FA-greedy, R-greedy with $R = 50$ and MD-greedy) and non-adaptive Haar-type wavelet dictionaries for two test functions (piecewise constant functions on Minnesota graph).

Image approximation with wedgelets

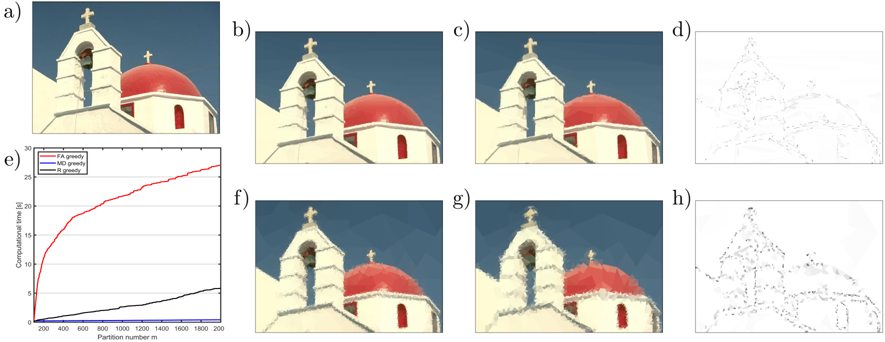
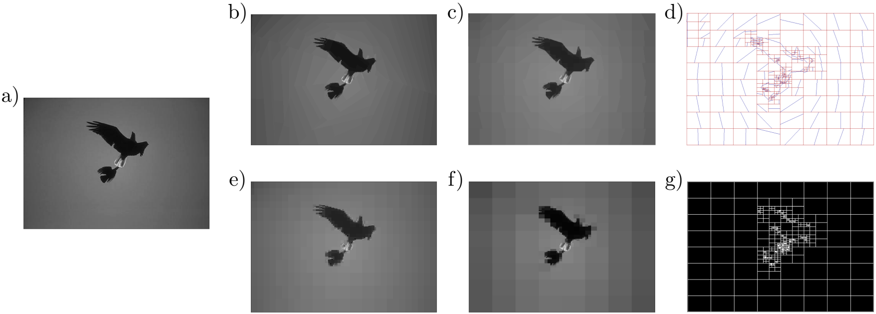


Fig. BWP image approximation.
a) original 481 × 321-image; b)c) FA-greedy BWP compression for $M = 2000$, $M = 1000$;
d) wavelet details between b) and c); e) Computational times of the BWP variants;
f)g) MD-greedy compression for $M = 2000$, $M = 1000$; h) wavelet details between f) & g).

Comparison to continuous wedgelets



Comparison of 4 image approximation techniques based on piecewise approximation:
a) original 481 × 321 image; b) graph wedgelet compression using 500 most relevant BWP wavelet coefficients (PSNR: 38.297 dB) c)d) continuous wedgelet compression using 506 wedges (PSNR: 36.828 dB, code by F. Friedrich) e) Haar wavelet compression using 500 most relevant coefficients (PSNR: 34.764 dB) f)g) quadtree compression with 505 blocks (PSNR: 31.662 dB, intern Matlab implementation).

Memory requirements for a wedgelet compression

Beside the quantized mean values $\bar{f}_{V_{\mathrm{q}_i}^{(M)}}$, we also have to store the BWP tree $\mathcal{T}_Q$ in terms of the node set Q.

Theorem 5

Assume that the mean values $\bar{f}_{V_{\mathrm{q}_i}^{(M)}}$ are given in a quantized form with at most K different values. Then, the wedgelet encoding in Algorithm 1 requires a memory of at most

$$\frac{\lceil \log_2(n) + \log_2(K) \rceil M}{n} \quad \textit{bits per node.}$$

Example: In the particular case of an image with $512 \times 512 = 2^{18}$ pixels and an image depth of $K = 2^8 = 256$ colors we get by Theorem 5 that a representation with $M = 1000$ wedgelets requires a memory of less than 0.1 bits per pixel.

Geometric wavelets related to graph wedgelets

Instead of storing mean values $\{\bar{f}_{V^{(M)}_{\mathrm{q}_1}}, \ldots, \bar{f}_{V^{(M)}_{\mathrm{q}_M}}\}$, we can alternatively encode $\mathcal{W}_M f$ using geometric wavelets w.r.t the BWP tree $\mathcal{T}_Q$.

Let $W', W \in \mathcal{T}_Q$ such that W' is a child of W. Then, the wavelet component $\psi_{W'}(f)$ is given as (Dekel, Leviatan [4])

$$\psi_{W'}(f)(\mathrm{v}) = \left(\frac{\langle f, \chi_{W'} \rangle}{|W'|} - \frac{\langle f, \chi_W \rangle}{|W|} \right) \chi_{W'}(\mathrm{v}). \tag{3}$$

In this way, we obtain for every child W' in $\mathcal{T}_Q$ a wavelet component $\psi_{W'}(f)$ of f. For the root $V \in \mathcal{T}$, we set $\psi_V(f)(\mathrm{v}) = \frac{\langle f, \chi_V \rangle}{|V|}$.

Using geometric wavelets as a description for the wedgelet approximation is particularly suited if a further compression of f is desired, for instance by using an m-term approximation of f with $m < M$.

m-term approximation with geometric wavelets

To see whether a graph signal f can be approximated sparsely by piecewise constant functions on the elements of a BWP tree, the $\mathcal{L}^2$-error $\|f - \mathcal{S}_m(f)\|_{\mathcal{L}^2(V)}$ can be analyzed, where $\mathcal{S}_m(f)$ denotes the best m-term approximation

$$\mathcal{S}_m(f)(\mathrm{v}) = \sum_{i=1}^{m} \psi_{W_i}(f)(\mathrm{v}) \tag{4}$$

of f w.r.t. m wavelets $\psi_{W_i}(f)$, $i \in \{1, \ldots, m\}$. These Haar-type wavelets are sorted descendingly in terms of the $\mathcal{L}^2$-norm:

$$\|\psi_{W_1}(f)\|_{\mathcal{L}^2(V)} \geq \|\psi_{W_2}(f)\|_{\mathcal{L}^2(V)} \geq \|\psi_{W_3}(f)\|_{\mathcal{L}^2(V)} \geq \cdots .$$

Picking the m components with the largest $\mathcal{L}^2$-norm, we obtain the best non-linear m-term approximation $\mathcal{S}_m(f)$ of f.

m-term approximation with geometric wavelets

To study m-term approximation the following energy functional is of main relevance (see, for instance, Devore). It is the discrete counterpart of a functional given by Dekel & Leviatan for binary space partitionings in hypercubes.

Definition 6

For $0 < r < \infty$, we define the r-energy of the wavelet components of f with respect to a BWP tree $\mathcal{T}_Q$ as

$$\mathcal{N}_r(f, \mathcal{T}_Q) = \left(\sum_{W \in \mathcal{T}_Q} \|\psi_W(f)\|^r_{\mathcal{L}^2(V)} \right)^{\frac{1}{r}}.$$

For wavelets, this functional is used in the characterization of Besov spaces and measures the sparseness of the wavelet representation of a signal f.

Similar as the r-energy functional $\mathcal{N}_r(f, \mathcal{T}_Q)$, the next Besov-type smoothness term quantifies how well f can be approximated with piecewise constant functions on a BWP tree.

Definition 7

For $\alpha > 0$, $\frac{1}{2} \leq \rho < 1$, and $0 < r < \infty$, we define the geometric Besov-type smoothness measure $|\cdot|_{\mathcal{GB}_r^\alpha}$ as

$$|f|_{\mathcal{GB}_r^\alpha} = \inf_{\mathcal{T} \in \mathrm{BWP}} \left(\sum_{W \in \mathcal{T}} |W|^{-\alpha r} \sup_{\mathrm{w} \in W} \sum_{\mathrm{v} \in W} |f(\mathrm{v}) - f(\mathrm{w})|^r \right)^{\frac{1}{r}}.$$

In Dekel & Leviatan and Karaivanov & Petrushev, the corresponding spaces of functions have been referred to as geometric B-spaces. $|f|_{\mathcal{GB}_r^\alpha}$ is not linked to one particular BWP tree but allows to quantify the sparseness of f w.r.t. a largy family of BWP trees.

Theorem 8

Let $\alpha > 0$, $\frac{1}{2} \leq \rho < 1$ *and* $1/r = \alpha + 1/2$. *Further, let* $\mathcal{T}_Q(f)$ *be a near best* BWP *tree, i.e.,*

$$\mathcal{N}_r(f, \mathcal{T}_Q(f)) \leq C \inf_{\mathcal{T} \in \mathrm{BWP}} \mathcal{N}_r(f, \mathcal{T}).$$

Then, we have the equivalences

$$C_1 \mathcal{N}_r(f, \mathcal{T}_Q(f)) \leq |f|_{\mathcal{GB}_r^\alpha} \leq C_2 \mathcal{N}_r(f, \mathcal{T}_Q(f)).$$

with constants C_1 *and* C_2 *that depend only on* α *and* ρ. *Further,*

$$\|f - \mathcal{S}_m(f)\|_{\mathcal{L}^2(V)} \leq C m^{-\alpha} |f|_{\mathcal{GB}_r^\alpha}.$$

The constants $C_1, C_2, C > 0$ *depend only on* r *and* ρ.

The proof is largely based on the works of Dekel & Leviatan and Karaivanov & Petrushev.

Literature

Some of my work related to this talk:

[1] Erb, W. Graph Wedgelets: Adaptive Data Compression on Graphs based on Binary Wedge Partitioning Trees and Geometric Wavelets. *IEEE Trans. Signal Inf. Process. Netw. 9* (2023), 24-34.

[2] Cavoretto, R., De Rossi, A., and Erb, W. Partition of Unity Methods for Signal Processing on Graphs. *J. Fourier Anal. Appl. 27* (2021), Art. 66.

Software for graph wedgelets and geometric wavelets for images

`https://github.com/WolfgangErb/GraphWedgelets`

Thanks a lot for the invitation!