Wavelets and Filter Banks on Graphs

Pierre Vandergheynst
Signal Processing Lab, EPFL
Joint work with David Shuman

Duke Workshop on Sensing and Analysis of High-Dimensional Data
Duke University, July 2011
Processing Signals on Graphs

United States transmission grid
Source: FEMA

Social Network

Electrical Network

“Neuronal” Network

Transportation Network
Processing Signals on Graphs

United States transmission grid
Source: FEMA

Electrical Network

Social Network

“Neuronal” Network

Transportation Network
Short outline

- Summary of one wavelet construction on graphs
  - multiscale, filtering
- Pyramidal algorithms
  - polyphase components and downsampling
  - the Laplacian Pyramid
  - 2-channels, critically sampled filter banks
Spectral Graph Wavelets

\( G = (E, V) \) a weighted undirected graph, with Laplacian \( \mathcal{L} = D - A \)
Spectral Graph Wavelets

\[ G = (E, V) \] a weighted undirected graph, with Laplacian \( \mathcal{L} = D - A \)

Dilation operates through operator: \( T_g^t = g(t\mathcal{L}) \)
Spectral Graph Wavelets

\( G = (E, V) \) a weighted undirected graph, with Laplacian \( \mathcal{L} = D - A \)

**Dilation** operates through operator: \( T^t_g = g(t \mathcal{L}) \)

**Translation** (localization):

Define \( \psi_{t,j} = T^t_g \delta_j \) response to a delta at vertex \( j \)

\[
\psi_{t,j}(i) = \sum_{\ell=0}^{N-1} g(t \lambda_{\ell}) \phi_{\ell}^*(j) \phi_{\ell}(i) \quad \mathcal{L} \phi_{\ell}(j) = \lambda_{\ell} \phi_{\ell}(j)
\]

\[
\psi_{t,a}(u) = \int_{\mathbb{R}} d\omega \hat{\psi}(t\omega)e^{-j\omega a}e^{j\omega u}
\]
Spectral Graph Wavelets

$G = (E, V)$ a weighted undirected graph, with Laplacian $\mathcal{L} = D - A$

**Dilation** operates through operator: $T_g^t = g(tL)$

**Translation** (localization):

Define $\psi_{t,j} = T_g^t \delta_j$ response to a delta at vertex $j$

$$\psi_{t,j}(i) = \sum_{\ell=0}^{N-1} g(t\lambda_\ell) \phi^*_{\ell}(j) \phi_{\ell}(i) \quad \mathcal{L}\phi_{\ell}(j) = \lambda_\ell \phi_{\ell}(j)$$

$$\psi_{t,a}(u) = \int_{\mathbb{R}} d\omega \hat{\psi}(t\omega) e^{-j\omega a} e^{j\omega u}$$

And so formally define the graph wavelet coefficients of $f$:

$$W_f(t, j) = \langle \psi_{t,j}, f \rangle$$

$$W_f(t, j) = T_g^t f(j) = \sum_{\ell=0}^{N-1} g(t\lambda_\ell) \hat{f}(\ell) \phi_{\ell}(j)$$
Frames

\[ \exists A, B > 0, \exists h : \mathbb{R}_+ \to \mathbb{R}_+ \text{ (i.e. scaling function)} \]

\[ 0 < A \leq h^2(u) + \sum_s g(t_s u)^2 \leq B < \infty \]

\[ \phi_n = T_h \delta_n = h(L) \delta_n \]

A simple way to get a tight frame:

\[ \gamma(\lambda) = \int_{1/2}^{1} \frac{dt}{t} g^2(t\lambda) \quad \Rightarrow \quad \tilde{g}(\lambda) = \sqrt{\gamma(\lambda) - \gamma(2\lambda)} \]

for any admissible kernel \( g \)
Scaling & Localization

\[ \psi_{t,i}(j) \]
Scaling & Localization

$\psi_{t,i}(j)$

decreasing scale
Example
Example
Example
Example
Example
Example
Sparsity and Smoothness on Graphs

scaling functions coeffs

Sensing and Analysis of High-D Data
Duke University July 2011
Remark on Implementation

Not necessary to compute spectral decomposition for filtering

Polynomial approximation: \( g(t\omega) \approx \sum_{k=0}^{K-1} a_k(t)p_k(\omega) \)

ex: Chebyshev, minimax

Then wavelet operator expressed with powers of Laplacian:

\[ T_g^t \approx \sum_{k=0}^{K-1} a_k(t)\mathcal{L}^k \]

And use sparsity of Laplacian in an iterative way
Remark on Implementation

\[ \tilde{W}_f(t, j) = (\rho(\mathcal{L}) f^\#)_j \quad |W_f(t, j) - \tilde{W}_f(t, j)| \leq B \|f\| \]

sup norm control (minimax or Chebyshev)

\[ \tilde{W}_f(t_n, j) = \left( \frac{1}{2} c_{n,0} f^\# + \sum_{k=1}^{M_n} c_{n,k} \overline{T}_k(\mathcal{L}) f^\# \right)_j \]

\[ \overline{T}_k(\mathcal{L}) f = \frac{2}{a_1} (\mathcal{L} - a_2 I) (\overline{T}_{k-1}(\mathcal{L}) f) - \overline{T}_{k-2}(\mathcal{L}) f \]

Computational cost dominated by matrix-vector multiply with (sparse) Laplacian matrix.
In particular \( O(\sum_{n=1} M_n |E|) \)

Note: “same” algorithm for adjoint!

http://wiki.epfl.ch/sgwt
Graph wavelets

• Redundancy breaks sparsity
  - can we remove some or all of it?

• Faster algorithms
  - traditional wavelets have fast filter banks implementation
  - whatever scale, you use the same filters
  - here: large scales -> more computations

• Goal: solve both problems at one
Basic Ingredients

Euclidean multiresolution is based on two main operations

Filtering (typically low-pass and high-pass)
Down and Up sampling
Basic Ingredients

Euclidean multiresolution is based on two main operations

Filtering (typically low-pass and high-pass)
Down and Up sampling
Basic Ingredients

Euclidean multiresolution is based on two main operations

Filtering (typically low-pass and high-pass)
Down and Up sampling

Filtering is fine but how do we downsample on graphs ??
Basic Ingredients

Subsampling is equivalent to splitting in two cosets (even, odd)

\[ \bullet \bullet \bullet \bullet \bullet \bullet \bullet \bullet \]

\[ \bullet \ \bigcirc \ \bullet \ \bigcirc \ \bullet \ \bigcirc \ \bullet \ \bigcirc \ ]
Basic Ingredients

Subsampling is equivalent to splitting in two cosets (even, odd)

Questions:
- How do we partition a graph into meaningful cosets?
- Are there efficient algorithms for these partitions?
- Are there theoretical guarantees?
- How do we define a new graph from the cosets?
Cosets - A spectral view

Subsampling is equivalent to splitting in two cosets (even, odd)

\[ f_{sub}(i) = \frac{1}{2} f(i)(1 + \cos(\pi i)) \]

Classically, selecting a coset can be interpreted easily in Fourier:

- eigenvector of largest eigenvalue
Cosets and Nodal Domains

Nodal domain: maximally connected subgraph s.t. all vertices have same sign w.r.t a reference function

We would like to find a very large number of nodal domains, ideally $|V|!$

Nodal domains of Laplacian eigenvectors are special (and well studied)
Cosets and Nodal Domains

Nodal domain: maximally connected subgraph s.t. all vertices have same sign w.r.t. a reference function

We would like to find a very large number of nodal domains, ideally $|V|$!

Nodal domains of Laplacian eigenvectors are special (and well studied)

**Theorem:** the number of nodal domains associated to the largest laplacian eigenvector of a connected graph is maximal,

$$\nu(\phi_{\text{max}}) = \nu(G) = |V|$$

IFF $G$ is bipartite

In general: $\nu(G) = |V| - \chi(G) + 2$ (extreme cases: bipartite and complete graphs)
Cosets and Nodal Domains

Nodal domain: maximally connected subgraph s.t. all vertices have same sign w.r.t a reference function

We would like to find a very large number of nodal domains, ideally \(|V|\)!

Nodal domains of Laplacian eigenvectors are special (and well studied)

For any connected graph we will thus naturally define cosets and their associated selection functions

\[
V_+ = \{ i \in V \text{ s.t. } \phi_{N-1}(i) \geq 0 \} \\
M_+(i) = \frac{1}{2} (1 + \text{sgn}(\phi_{N-1}(i))) \\
V_- = \{ i \in V \text{ s.t. } \phi_{N-1}(i) < 0 \} \\
M_-(i) = \frac{1}{2} (1 - \text{sgn}(\phi_{N-1}(i)))
\]
Examples of cosets

Simple line graph

\[ \phi_k(u) = \sin\left(\frac{\pi ku}{n} + \frac{\pi}{2n}\right) \quad \lambda_k = 2 - 2 \cos\left(\frac{\pi k}{n}\right) \quad 1 \leq k \leq n \]
Examples of cosets

Simple line graph

\[ \phi_k(u) = \sin\left(\frac{\pi ku}{n} + \frac{\pi}{2n}\right) \quad \lambda_k = 2 - 2 \cos\left(\frac{\pi k}{n}\right) \quad 1 \leq k \leq n \]
Examples of cosets

Simple line graph

Simple ring graph

\[ \phi_1^k(u) = \sin\left(\frac{2\pi k u}{n}\right) \quad \phi_2^k(u) = \cos\left(\frac{2\pi k u}{n}\right) \quad 1 \leq k \leq n/2 \]

\[ \lambda_k = 2 - 2 \cos\left(\frac{2\pi k}{n}\right) \]
Examples of cosets

Simple line graph

\[ \phi^1_k(u) = \sin\left(\frac{2\pi ku}{n}\right) \quad \phi^2_k(u) = \cos\left(\frac{2\pi ku}{n}\right) \quad 1 \leq k \leq n/2 \]

\[ \lambda_k = 2 - 2 \cos\left(\frac{2\pi k}{n}\right) \]
Examples of cosets

Simple line graph

Simple ring graph

Lattice
Examples of cosets

Simple line graph

Simple ring graph

Lattice

quincunx
The Agonizing Limits of Intuition

- Multiplicity of $\lambda_{\text{max}}$
  - how do we choose the control vector in that subspace?
  - even a prescription can be numerically ill-defined
  - graphs with “flat” spectrum in close to their spectral radius

- Laplacian eigenvectors do not always behave like global oscillations
  - seems to be true for random perturbations of simple graphs
  - true even for a class of trees [Saito2011]
The Laplacian Pyramid

Analysis operator

\[ x \rightarrow H \rightarrow G \rightarrow U \rightarrow y_{low} \]

\[ y_1 \rightarrow - \]
The Laplacian Pyramid

Analysis operator

\[ x \rightarrow \begin{array}{c}
H \\
D \\
U \\
\end{array} \rightarrow \begin{array}{c}
y_0 \\
\end{array} \]

\[ \begin{array}{c}
G \\
\end{array} \rightarrow \begin{array}{c}
y_1 \\
\end{array} \]
The Laplacian Pyramid

Analysis operator

\[ x \rightarrow H \rightarrow \text{G} \rightarrow \text{M} \rightarrow y_0 \]

\[ y_1 \rightarrow \text{G} \rightarrow \text{M} \rightarrow y_0 \]
The Laplacian Pyramid

Analysis operator

\[ y_0 = H_m x \]
\[ = MHx \]

\[ y_1 = x - G y_0 \]
\[ = x - GH_m x \]
The Laplacian Pyramid

Analysis operator

\[ x \xrightarrow{\text{H}} \xrightarrow{\text{M}} \xrightarrow{\text{G}} y_0 \xrightarrow{\text{y}} y_1 \]
The Laplacian Pyramid

Analysis operator

\[ \begin{pmatrix} y_0 \\ y_1 \\ y \end{pmatrix} = \begin{pmatrix} H_m & I - GH_m & T_a \end{pmatrix} x, \]

\[ y_0 = H M x \]

\[ y_1 = x - G y_0 \]

\[ y_0 = H M x = MHx = MV \tilde{H} V^T x, \]

where \( V = [v_0 | v_1 | ... | v_{n-1}] \) is the matrix of the eigenvectors of the graph Laplacian and \( \tilde{H} \) is a diagonal matrix with \( \lambda \) entries on the diagonal and zeros elsewhere.

\( \text{upsampling by masking operator} \)

\( M \) is a diagonal matrix with ones at the diagonal entries corresponding to the location of the selected vertices \( v \) and zeros elsewhere.

Then we will pass the output of the masking block through a second filter \( g \) in order to reconstruct the original function \( x \). Finally, the reconstruction error is easily computed by taking the difference of the original signal and the output of the second filter.

Consider an input graph \( x \) signal \( x \in \mathbb{R}^n \). In our notation, \( y_0 = H M x \) denotes the output of the first stage followed by the masking operator. This is the output of the lowpass channel in the LP framework.
The Laplacian Pyramid

Analysis operator

\[
\begin{pmatrix}
    y_0 \\
y_1
\end{pmatrix}
= \begin{pmatrix}
    H_m \\
    I - GH_m
\end{pmatrix}
\begin{pmatrix}
    x
\end{pmatrix},
\]

\[T_a\]
The Laplacian Pyramid

Analysis operator

\[
\begin{pmatrix}
y_0 \\
y_1 \\
y
\end{pmatrix} = \begin{pmatrix}
H_m \\
I - GH_m
\end{pmatrix}_T a x,
\]

Simple (traditional) left inverse

\[
\hat{x} = \begin{pmatrix}
G & I
\end{pmatrix}_T s \begin{pmatrix}
y_0 \\
y_1 \\
y
\end{pmatrix}
\]

\[T_s T_a = I \quad \text{with no conditions on } H \text{ or } G\]
The Laplacian Pyramid

Pseudo Inverse?

\[ T_a^\dagger = (T_a^T T_a)^{-1} T_a^T \]

Let’s try to use only filters
The Laplacian Pyramid

Pseudo Inverse ?

$T_a^\dagger = (T_a^T T_a)^{-1} T_a^T$

Let’s try to use only filters

Define iteratively, through descent on LS:

$\arg \min_x \|T_a x - y\|_2^2 \quad \Rightarrow \quad \hat{x}_{k+1} = \hat{x}_k + \tau T_a^T (y - T_a \hat{x}_k)$

$T_a^T = (H_m^T \quad I - H_m^T G^T)$
The Laplacian Pyramid

we can easily implement $T_a^T T_a$ with filters and masks:

$$x_N = \tau \sum_{j=0}^{N-1} (I - \tau Q)^j b$$

Use Chebyshev approximation of:

$$L(\omega) = \tau \sum_{j=0}^{N-1} (1 - \tau \omega)^j$$
Kron Reduction

In order to iterate the construction, we need to construct a graph on the reduced vertex set.

\[ A_r = A[\alpha, \alpha] - A[\alpha, \alpha]A(\alpha, \alpha)^{-1}A(\alpha, \alpha) \]

\[ A = \begin{bmatrix} A[\alpha, \alpha] & A[\alpha, \alpha] \\ A(\alpha, \alpha) & A(\alpha, \alpha) \end{bmatrix} \]
In order to iterate the construction, we need to construct a graph on the reduced vertex set.

\[
A_r = A[\alpha, \alpha] - A[\alpha, \alpha]A(\alpha, \alpha)^{-1}A(\alpha, \alpha)
\]

\[
A = \begin{bmatrix}
A[\alpha, \alpha] & A[\alpha, \alpha] \\
A(\alpha, \alpha) & A(\alpha, \alpha)
\end{bmatrix}
\]

Kron Reduction

[Dorfler et al, 2011]
Kron Reduction

In order to iterate the construction, we need to construct a graph on the reduced vertex set.

\[ A_r = A[\alpha, \alpha] - A[\alpha, \alpha]A(\alpha, \alpha)^{-1}A(\alpha, \alpha) \]

\[ A = \begin{bmatrix} A[\alpha, \alpha] & A[\alpha, \alpha] \\ A(\alpha, \alpha) & A(\alpha, \alpha) \end{bmatrix} \]

Properties:
- maps a weighted undirected laplacian to a weighted undirected laplacian
- spectral interlacing (spectrum does not degenerate)
  \[ \lambda_k(A) \leq \lambda_k(A_r) \leq \lambda_{k+n-|\alpha|}(A) \]
- disconnected vertices linked in reduced graph IFF there is a path that runs only through eliminated nodes
Example

Note: For a k-regular bipartite graph

\[ L = \begin{bmatrix} kI_n & -A \\ -A^T & kI_n \end{bmatrix} \]

Kron-reduced Laplacian: \( L_r = k^2I_n - AA^T \)
Note: For a k-regular bipartite graph

\[
L = \begin{bmatrix}
kI_n & -A \\
-A^T & kI_n
\end{bmatrix}
\]

Kron-reduced Laplacian: \( L_r = k^2I_n - AA^T \)

\[
\hat{f}_r(i) = \hat{f}(i) + \hat{f}(N - i) \quad i = 1, ..., N/2
\]
Filter Banks

2 critically sampled channels

\[
\begin{array}{c}
\text{Coset 1} \\
\text{Coset 2}
\end{array}
\]

\[
\begin{array}{c}
\text{Filter } H \\
\text{Filter } G
\end{array}
\]

\[
\begin{array}{c}
\text{Downsample} \\
\text{Downsample}
\end{array}
\]

\[
f_0
\]
Filter Banks

2 critically sampled channels

\[ f_0 \]

\[ \begin{align*}
\text{Filter } H & \quad \text{Downsample} \\
\text{Coset 1} & \\
\text{Filter } G & \quad \text{Downsample} \\
\text{Coset 2} &
\end{align*} \]

**Theorem:** For a k-RBG, the filter bank is perfect-reconstruction IFF

\[ |H(i)|^2 + |G(i)|^2 = 2 \]

\[ H(i)G(N - i) + H(N - i)G(i) = 0 \]
Conclusions

• Structured, data dependent dictionary of wavelets
  - sparsity and smoothness on graph are merged in simple and elegant fashion
  - fast algo, clean problem formulation
  - graph structure can be totally hidden in wavelets

• Filter banks based on nodal domains or coloring
  - Universal algo based on filtering and Kron reduction
  - Efficient IFF *some* structure in the graph
  - Unfortunately no closed form theory in general
Wavelet Ingredients

Wavelet transform based on two operations:

**Dilation** (or scaling) and **Translation** (or localization)

\[ \psi_{s,a}(x) = \frac{1}{s} \psi \left( \frac{x - a}{s} \right) \]
Wavelet Ingredients

Wavelet transform based on two operations:

Dilation (or scaling) and Translation (or localization)

$$\psi_{s,a}(x) = \frac{1}{s} \psi \left( \frac{x - a}{s} \right)$$

$$(T^s f)(a) = \int \frac{1}{s} \psi^* \left( \frac{x - a}{s} \right) f(x) dx \quad (T^s f)(a) = \langle \psi_{s,a}, f \rangle$$
Wavelet Ingredients

Wavelet transform based on two operations:

**Dilation** (or scaling) and **Translation** (or localization)

\[
\psi_{s,a}(x) = \frac{1}{s} \psi \left( \frac{x - a}{s} \right)
\]

\[
(T^s f)(a) = \int \frac{1}{s} \psi^* \left( \frac{x - a}{s} \right) f(x) dx \quad (T^s f)(a) = \langle \psi_{s,a}, f \rangle
\]

Equivalently:

\[
(T^s \delta_a)(x) = \frac{1}{s} \psi^* \left( \frac{x - a}{s} \right)
\]

\[
(T^s f)(x) = \frac{1}{2\pi} \int e^{i\omega x} \hat{\psi}^*(s\omega) \hat{f}(\omega) d\omega
\]
Graph Laplacian and Spectral Theory

\[ G = (V, E, w) \] weighted, undirected graph

Non-normalized Laplacian: \[ \mathcal{L} = D - A \] Real, symmetric

\[ (\mathcal{L} f)(i) = \sum_{i \sim j} w_{i,j} (f(i) - f(j)) \]

Why Laplacian?
Graph Laplacian and Spectral Theory

\[ G = (V, E, w) \] weighted, undirected graph

Non-normalized Laplacian: \( \mathcal{L} = D - A \) Real, symmetric

\[
(\mathcal{L} f)(i) = \sum_{i \sim j} w_{i,j} (f(i) - f(j))
\]

Why Laplacian? \( \mathbb{Z}^2 \) with usual stencil

\[
(\mathcal{L} f)_{i,j} = 4f_{i,j} - f_{i+1,j} - f_{i-1,j} - f_{i,j+1} - f_{i,j-1}
\]

In general, graph laplacian from nicely sampled manifold converges to Laplace-Beltrami operator
Graph Laplacian and Spectral Theory

\[ \frac{d^2}{dx^2} \quad \Rightarrow \quad e^{i\omega x} \quad \Rightarrow \quad f(x) = \frac{1}{2\pi} \int \hat{f}(\omega) e^{i\omega x} \, d\omega \]
Graph Laplacian and Spectral Theory

\[ \frac{d^2}{dx^2} \quad \implies \quad e^{i\omega x} \quad \implies \quad f(x) = \frac{1}{2\pi} \int \hat{f}(\omega)e^{i\omega x} \, d\omega \]

Eigen decomposition of Laplacian: \( \mathcal{L}\phi_l = \lambda_l \phi_l \)
Graph Laplacian and Spectral Theory

\[ \frac{d^2}{dx^2} \rightarrow e^{i\omega x} \rightarrow f(x) = \frac{1}{2\pi} \int \hat{f}(\omega) e^{i\omega x} d\omega \]

Eigen decomposition of Laplacian: \( \mathcal{L}\phi_l = \lambda_l \phi_l \)

For simplicity assume connected graph and \( 0 = \lambda_0 < \lambda_1 \leq \lambda_2 \ldots \leq \lambda_{N-1} \)

For any function on the vertex set (vector) we have:

\[ \hat{f}(l) = \langle \phi_l, f \rangle = \sum_{i=1}^{N} \phi_l^*(i) f(i) \quad \text{Graph Fourier Transform} \]

\[ f(i) = \sum_{\ell=0}^{N-1} \hat{f}(\ell) \phi_\ell(i) \]
Spectral Graph Wavelets

Remember good old Euclidean case:

\[(T^s f)(x) = \frac{1}{2\pi} \int e^{i\omega x} \hat{\psi}^*(s\omega) \hat{f}(\omega) d\omega\]

We will adopt this operator view
Spectral Graph Wavelets

Remember good old Euclidean case:

\[
(T^s f)(x) = \frac{1}{2\pi} \int e^{i\omega x} \hat{\psi}^*(s\omega) \hat{f}(\omega) d\omega
\]

We will adopt this operator view

Operator-valued function via continuous Borel functional calculus

\[
g : \mathbb{R}^+ \rightarrow \mathbb{R}^+ \quad T_g = g(\mathcal{L}) \quad \text{Operator-valued function}
\]

Action of operator is induced by its Fourier symbol

\[
\hat{T}_g \hat{f}(\ell) = g(\lambda_\ell) \hat{f}(\ell) \quad (T_g f)(i) = \sum_{\ell=0}^{N-1} g(\lambda_\ell) \hat{f}(\ell) \phi_\ell(i)
\]
Non-local Wavelet Frame

- Non-local Wavelets are ...

... Graph Wavelets on Non-Local Graph

Interest: good *adaptive* sparsity basis
Distributed Computation

Scenario: Network of N nodes, each knows
- local data f(n)
- local neighbors
- M Chebyshev coefficients of wavelet kernel
- A global upper bound on largest eigenvalue of graph laplacian
Distributed Computation

Scenario: Network of N nodes, each knows
- local data f(n)
- local neighbors
- M Chebyshev coefficients of wavelet kernel
- A global upper bound on largest eigenvalue of graph laplacian

To compute: \( (\tilde{\Phi} f)_{(j-1)N+n} = \left( \frac{1}{2} c_{j,0} f + \sum_{k=1}^{M} c_{j,k} \overline{T}_k(L)f \right)_n \)
Distributed Computation

Scenario: Network of N nodes, each knows
- local data f(n)
- local neighbors
- M Chebyshev coefficients of wavelet kernel
- A global upper bound on largest eigenvalue of graph laplacian

To compute: \((\tilde{\Phi} f)_{(j-1)N+n} = \left(\frac{1}{2} c_{j,0} f + \sum_{k=1}^{M} c_{j,k} T_k(L) f\right)_n\)

\[\left(\overline{T}_1(L) f\right)_n = \left(\frac{2}{\alpha} (L - \alpha I) f\right)_n\]  
sensor only needs f(n) from its neighbors
Distributed Computation

**Scenario:** Network of N nodes, each knows
- local data $f(n)$
- local neighbors
- M Chebyshev coefficients of wavelet kernel
- A global upper bound on largest eigenvalue of graph laplacian

To compute: \[(\tilde{\Phi} f)_{(j-1)N+n} = (\frac{1}{2} c_{j,0} f + \sum_{k=1}^{M} c_{j,k} \overline{T}_k(\mathcal{L}) f)_{n}\]

\[\left(\overline{T}_1(\mathcal{L}) f\right)_{n} = \left(\frac{2}{\alpha} (\mathcal{L} - \alpha I) f\right)_{n}\]  
sensor only needs $f(n)$ from its neighbors

\[\left(\overline{T}_k(\mathcal{L}) f\right) = \frac{2}{\alpha} (\mathcal{L} - \alpha I) \left(\overline{T}_{k-1}(\mathcal{L}) f\right) - \overline{T}_{k-2}(\mathcal{L}) f\]  
Computed by exchanging last computed values
Distributed Computation

Communication cost: $2M|E|$ messages of length 1 per node

Example: distributed denoising, or distributed regression, with Lasso

$$\arg\min_a \frac{1}{2} \| y - \Phi^* a \|^2 + \|a\|_{1,\mu}$$

$$a_i^{(k)} = S_{\mu_i,\tau} \left( [a^{k-1} + \tau \Phi (y - \Phi^* a^{k-1})]_i \right)$$

$$S_{\mu_i\tau}(z) := \begin{cases} 0, & \text{if } |z| \leq \mu_i\tau \\ z - \text{sgn}(z)\mu_i\tau, & \text{o.w.} \end{cases}$$

Total communication cost:

Distributed Lasso [Mateos, Bazerque, Gianakis] \hspace{1cm} Cost $\sim |E|N$

Chebyshev $\Phi y$ \hspace{1cm} $2M|E|$ messages of length 1 \hspace{1cm} Cost $\sim |E|$

$\Phi \Phi^* a$ \hspace{1cm} $4M|E|$ messages of length $J+1$
Wavelets on Graphs?

- Existing constructions
  - wavelets on meshes (computer graphics, numerical analysis), often via lifting
  - diffusion wavelets [Maggioni, Coifman & others]
  - recently several other constructions based on “organizing” graph in a multiscale way [Gavish-Coifman]

- Goal
  - process signals on graphs
  - retain simplicity and signal processing flavor
  - algorithm to handle fairly large graphs
Scaling & Localization

Effect of operator dilation?
Scaling & Localization

Effect of operator dilation?
Effect of operator dilation?
Scaling & Localization

Effect of operator dilation?

Theorem: $d_G(i, j) > K$ and $g$ has $K$ vanishing derivatives at 0

$$\frac{|\psi_{t,j}(i)|}{\|\psi_{t,j}\|} \leq Dt \quad \text{for any } t \text{ smaller than a critical scale}$$

function of $d_G(i, j)$

Reason? At small scale, wavelet operator behaves like power of Laplacian