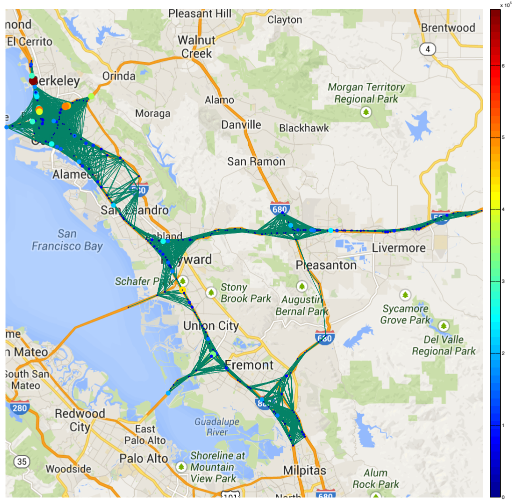
Spectral Graph Dictionaries

Pascal Frossard
EPFL

SAMPTA, May 2015

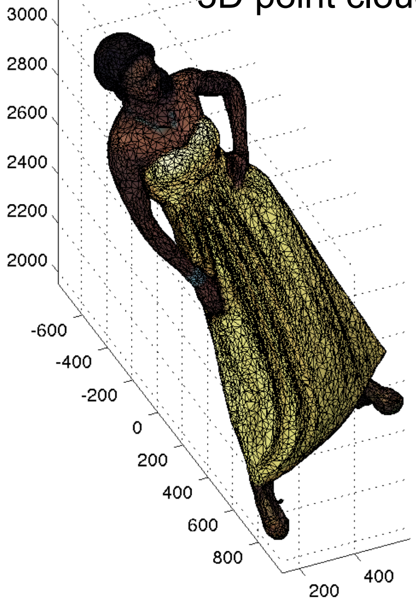
(Joint work with Dorina Thanou and David Shuman)

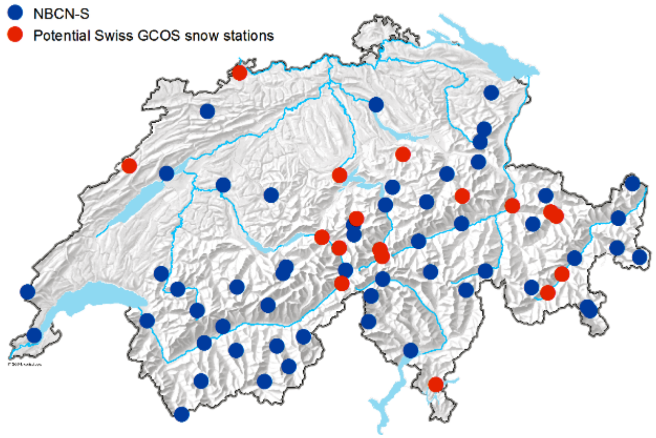
Structured data



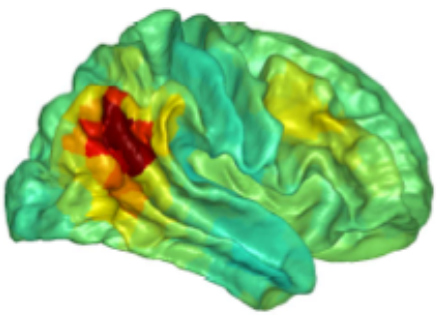
Traffic bottlenecks

3D point clouds

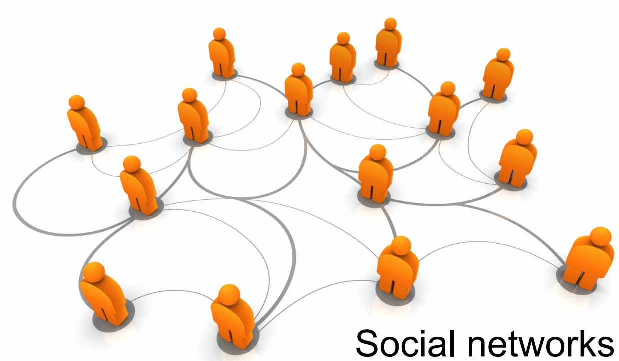




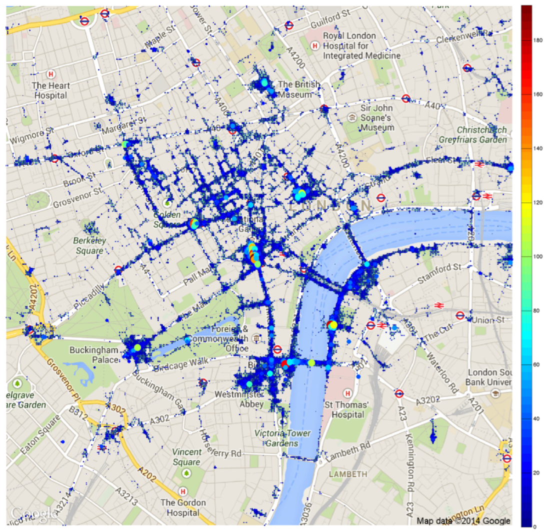
Sensor networks



Brain signals



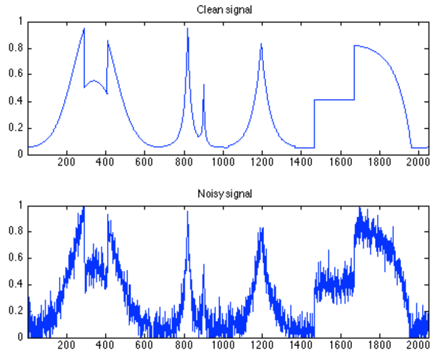
Social networks



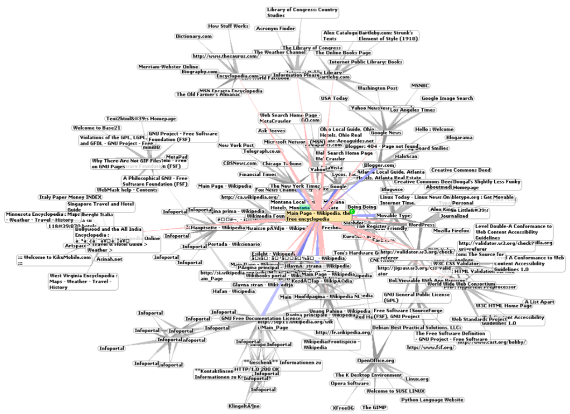
Mobility patterns

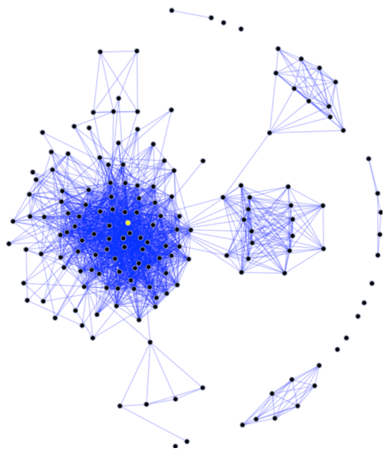
Structured, but irregular data …

- Traditional signal processing in Euclidean space



- Irregular (graph) structures: new challenges for signal processing?





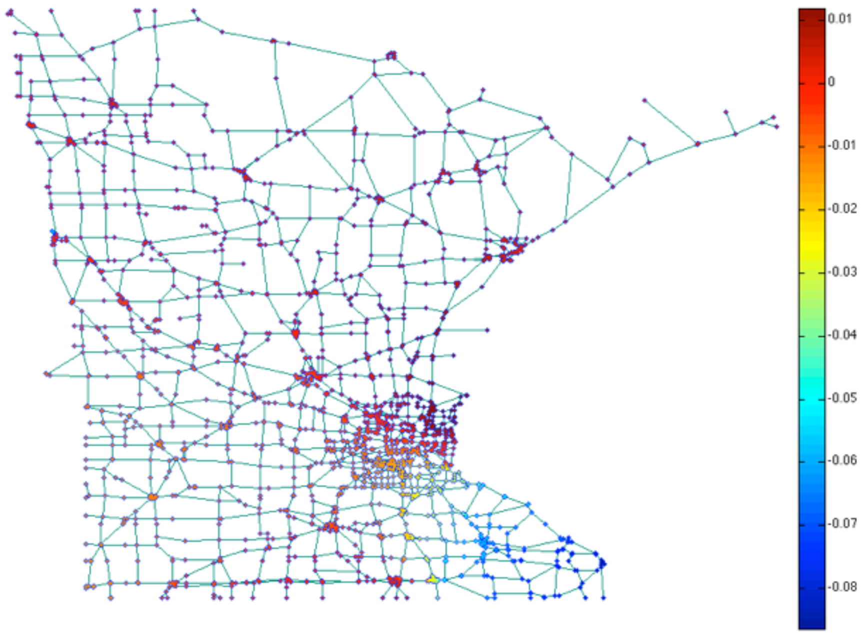
Challenges on graphs

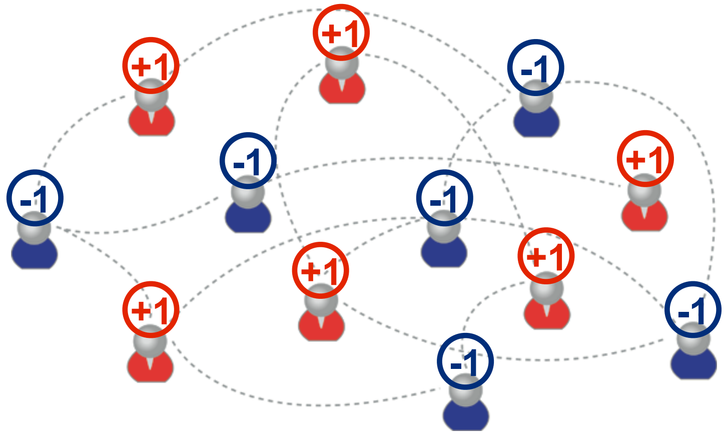
- Data processing on irregular domains raises important questions:
  - how to incorporate the graph structure into *localised* transforms?
  - how to leverage invaluable intuitions from Euclidian framework?
  - how to design computationally effective methods?

- Sparsity is very helpful in classical settings - and for graphs?
  - could we define sparse representations on graphs?
  - could we build efficient dictionaries adapted to graphs?

Signal Processing on Graphs

- Objective: to process, analyse, reconstruct signals that live on networks or irregular structures



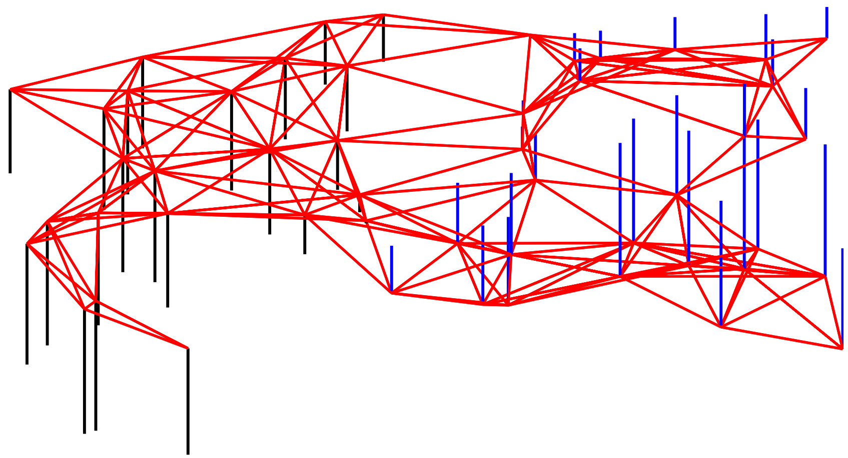


- Framework: emerging field of graph signal processing
  - algebraic and spectral graph theoretic concepts
  - harmonic analysis

Signals on Graphs

- Connected, undirected, weighted graph $\mathcal{G} = (V, E, W)$
  where $W_{i,j}$ is the weight of the edge $e = (i, j)$

- Graph signal: a function $f : \mathcal{V} \to \mathbb{R}$ that assigns real values to each vertex of the graph



- Graph description:
  - Degree matrix $\mathbf{D}$: diagonal matrix with sum of weights of incident edges
  - Laplacian matrix $\mathcal{L}$: difference operator defined based on $\mathbf{W}$

(Unormalized) Laplacian

- Laplacian is a difference operator $\mathcal{L} := \mathbf{D} - \mathbf{W}$

$$(\mathcal{L}f)(i) = \sum_{j \in \mathcal{N}_i} W_{i,j}[f(i) - f(j)]$$

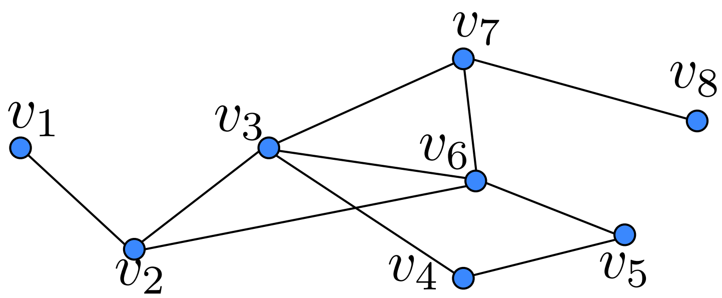
- It is a real symmetric matrix
- It has a complete set of eigenvectors $\{\mathbf{u}_\ell\}_{\ell=0,1,\ldots,N-1}$
- The eigenvectors are associated with real, nonnegative eigenvalues $\{\lambda_\ell\}_{\ell=0,1,\ldots,N-1}$

$$\mathcal{L}\mathbf{u}_\ell = \lambda_\ell \mathbf{u}_\ell, \ \forall \ell = 0, 1, \ldots, N-1$$

- Its spectrum is defined as $\sigma(\mathcal{L}) := \{\lambda_0, \lambda_1, \ldots, \lambda_{N-1}\}$

$$0 = \lambda_0 < \lambda_1 \leq \lambda_2 ... \leq \lambda_{N-1} := \lambda_{\max}$$

Laplacian example



$$G = \{V, E\}$$

$$D = diag(degree(v_1) \quad \ldots \quad degree(v_n))$$

$$\mathcal{L} := \mathbf{D} - \mathbf{W}$$

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & -1 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & -1 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$

$\mathbf{W}$ $\qquad$ $\mathcal{L}$

- Symmetric
- Off-diagonal entries non-positive
- Rows sum up to zero
- Has a complete set of orthonormal eigenvectors: $L = \chi \Lambda \chi^T$
  $0 = \lambda_0 < \lambda_1 \leq \ldots \leq \lambda_{n-1}$

Normalized Laplacian

- The normalized Laplacian is another popular graph matrix
- Each weight $W_{i,j}$ is normalised by $\dfrac{1}{\sqrt{d_i d_j}}$

$$\tilde{\mathcal{L}} := \mathbf{D}^{-\frac{1}{2}} \mathcal{L} \mathbf{D}^{-\frac{1}{2}}$$

$$(\tilde{\mathcal{L}} f)(i) = \frac{1}{\sqrt{d_i}} \sum_{j \in \mathcal{N}_i} W_{i,j} \left[\frac{f(i)}{\sqrt{d_i}} - \frac{f(j)}{\sqrt{d_j}} \right]$$

- The set of eigenvalues is $0 = \tilde{\lambda}_0 < \tilde{\lambda}_1 \leq \ldots \leq \tilde{\lambda}_{\max} \leq 2$
- The normalized Laplacian has often stability benefits

Graph Fourier Transform

- The eigenvectors of the graph Laplacian are used for defining the Graph Fourier Transform

GFT

$$\hat{f}(\lambda_\ell) := \langle \mathbf{f}, \mathbf{u}_\ell \rangle = \sum_{i=1}^{N} f(i) u_\ell^*(i)$$

IGFT

$$f(i) = \sum_{\ell=0}^{N-1} \hat{f}(\lambda_\ell) u_\ell(i)$$

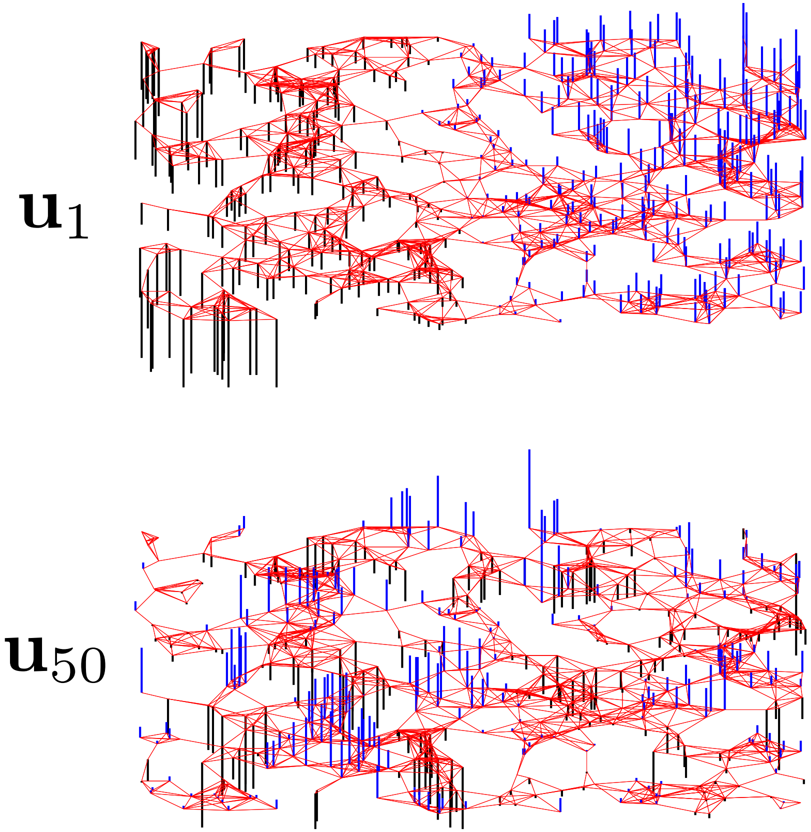
- This is analogous to the classical Fourier Transform built on eigenfunctions of the 1-D Laplace operator

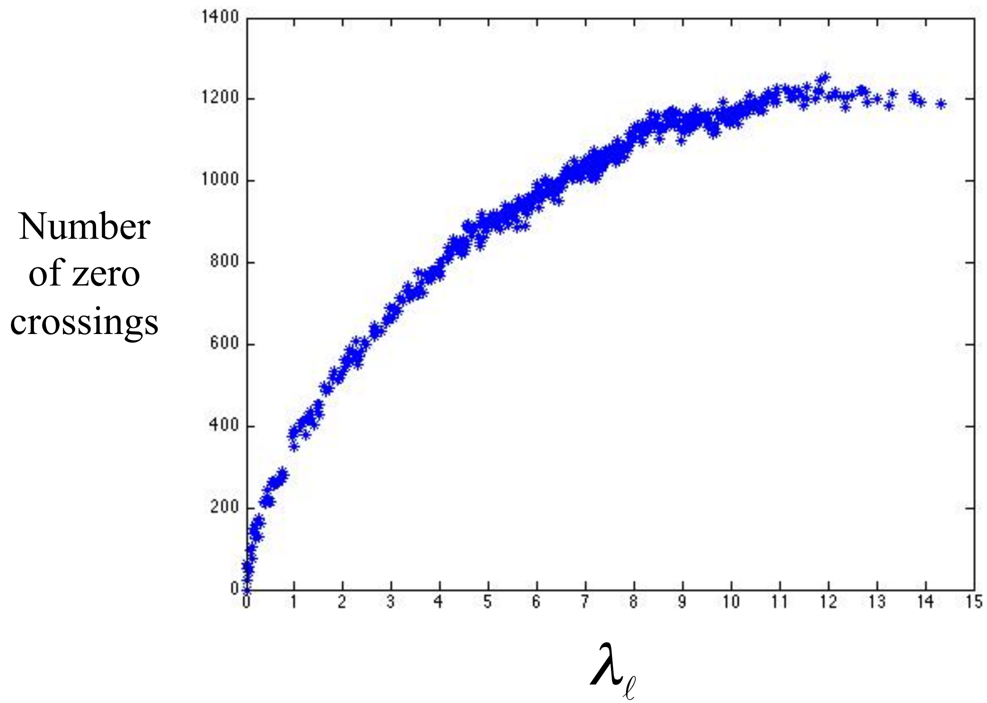
$$\hat{f}(\xi) := \langle f, e^{2\pi i \xi t} \rangle = \int_{\mathbb{R}} f(t) e^{-2\pi i \xi t} dt$$

$$-\Delta(e^{2\pi i \xi t}) = -\frac{\partial^2}{\partial t^2} e^{2\pi i \xi t} = (2\pi\xi)^2 e^{2\pi i \xi t}$$

Notion of 'frequency'

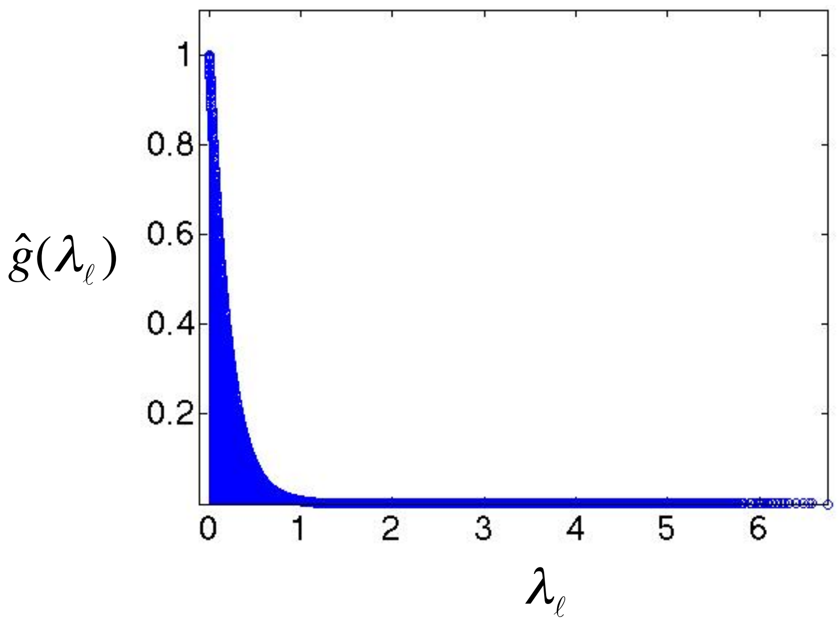
- The graph Laplacian eigenvalues and eigenvectors carry a notion of frequency



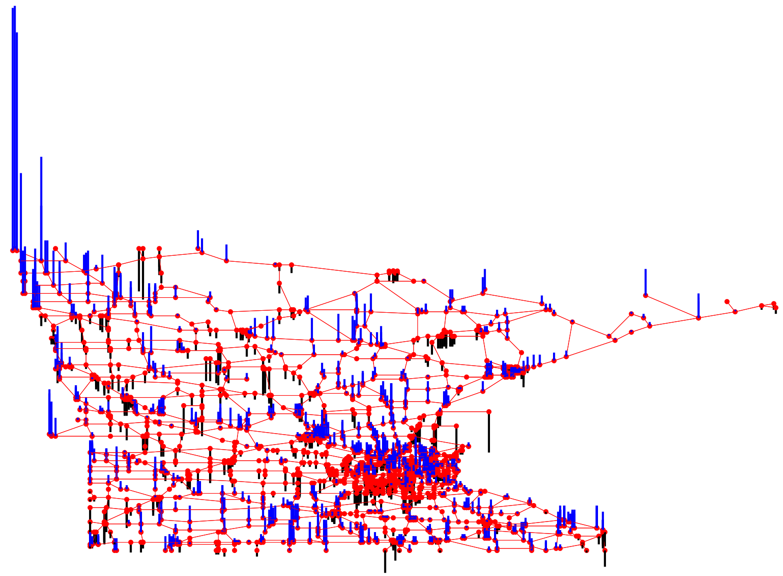


Dual representations

- Graph signals represented in either the vertex or the spectral domains (*kernels, or graph Fourier multipliers*)



$$\hat{g}(\lambda_\ell) = e^{-5\lambda_\ell}$$



$$g(n) \xleftarrow{IGFT} \hat{g}(\lambda_\ell)$$

Local Smoothness

- **Assumption**: strong interplay between signal and graph
  - Signal analysis driven by data structure
- Local smoothness at vertex i

$$\|\nabla_i \mathbf{f}\|_2 := \left[\sum_{j \in \mathcal{N}_i} W_{i,j} \left[f(j) - f(i) \right]^2 \right]^{\frac{1}{2}}$$

  - with the gradient $\nabla_i \mathbf{f} := \left[\left\{ \sqrt{W_{i,j}} \left[f(j) - f(i) \right] \right\}_{\text{for } j \in \mathcal{V} \text{ s.t. } e=(i,j) \in \mathcal{E}} \right]$

Global Smoothness

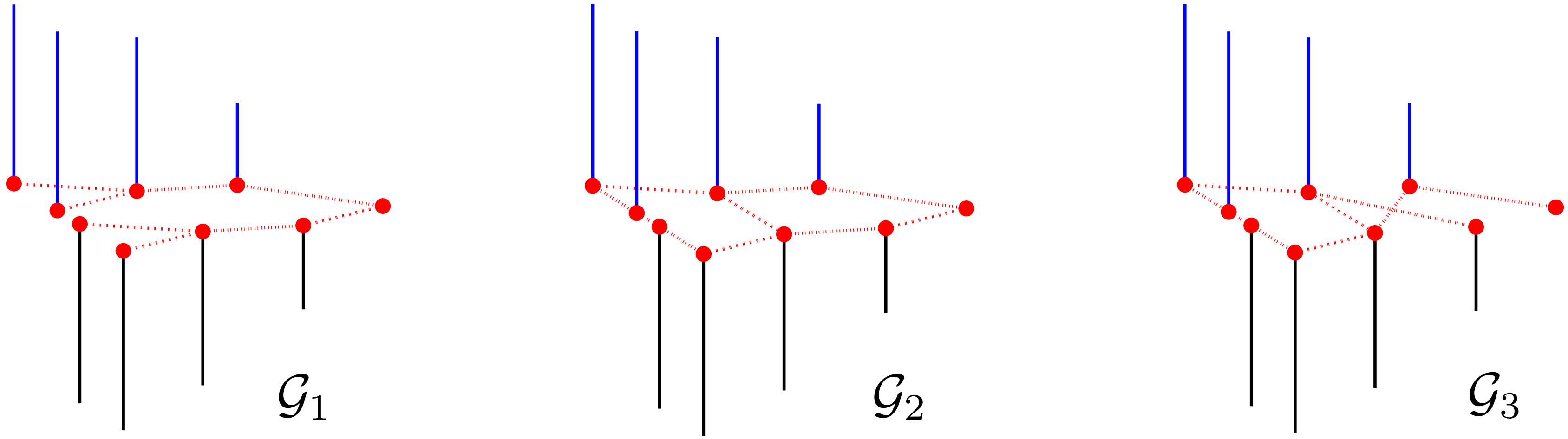
- **Assumption**: strong interplay between signal and graph
  - Signal analysis driven by data structure
- Global smoothness

$$S_p(\mathbf{f}) := \frac{1}{p} \sum_{i \in V} \|\nabla_i \mathbf{f}\|_2^p = \frac{1}{p} \sum_{i \in V} \left[\sum_{j \in \mathcal{N}_i} W_{i,j} \left[f(j) - f(i) \right]^2 \right]^{\frac{p}{2}}$$

  - with $p = 1$: total variation of the signal wrt the graph
  - with $p = 2$: graph Laplacian quadratic form

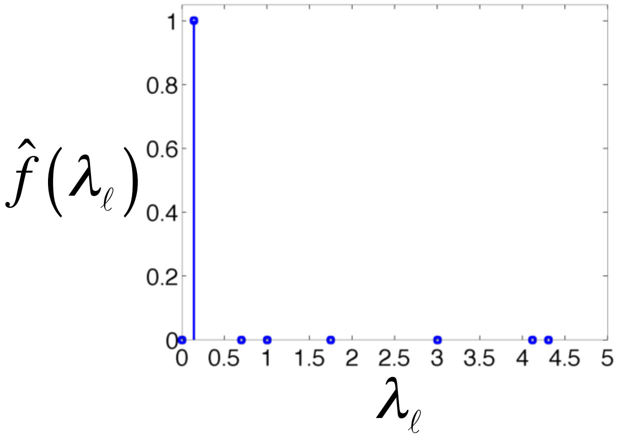
$$S_2(\mathbf{f}) = \frac{1}{2} \sum_{i \in V} \sum_{j \in \mathcal{N}_i} W_{i,j} \left[f(j) - f(i) \right]^2 = \sum_{(i,j) \in \mathcal{E}} W_{i,j} \left[f(j) - f(i) \right]^2 = \mathbf{f}^{\mathrm{T}} \mathcal{L} \mathbf{f}$$

Importance of the graph

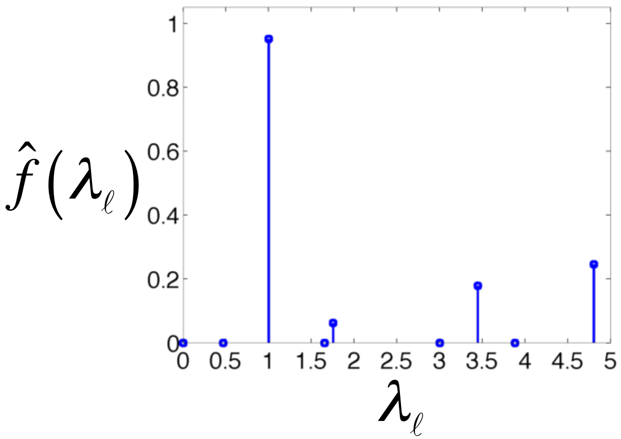


The same signal has different smoothness wrt different graphs

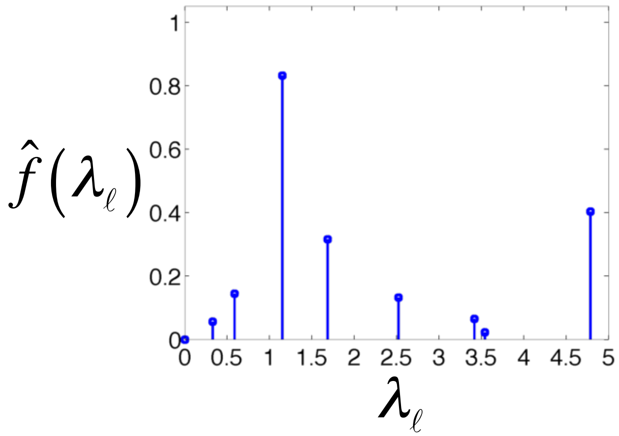
$\mathbf{f}^{\mathrm{T}}\mathcal{L}_1\mathbf{f} = 0.14$



$\mathbf{f}^{\mathrm{T}}\mathcal{L}_2\mathbf{f} = 1.31$



$\mathbf{f}^{\mathrm{T}}\mathcal{L}_3\mathbf{f} = 1.81$



Frequency filtering

- Analogously to classical filtering, one can perform graph spectral filtering with transfer function $\hat{h}(\lambda_\ell)$

$$\hat{f}_{out}(\lambda_\ell) = \hat{f}_{in}(\lambda_\ell)\hat{h}(\lambda_\ell)$$

- Equivalently $$f_{out}(i) = \sum_{\ell=0}^{N-1} \hat{f}_{in}(\lambda_\ell)\hat{h}(\lambda_\ell)u_\ell(i)$$

- In matrix notation: $$\mathbf{f}_{out} = \hat{h}(\mathcal{L})\mathbf{f}_{in}$$

$$\hat{h}(\mathcal{L}) := \mathbf{U} \begin{bmatrix} \hat{h}(\lambda_0) & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \hat{h}(\lambda_{N-1}) \end{bmatrix} \mathbf{U}^{\mathrm{T}}$$

Example: Tikhonov regularization

- Consider a classical denoising problem
  - noisy signal $\mathbf{y} = \mathbf{f}_0 + \boldsymbol{\eta}$
  - smooth regularization prior $\mathbf{f}^{\mathrm{T}} \mathcal{L} \mathbf{f}$
  - optimization problem: $\underset{\mathbf{f}}{\operatorname{argmin}} \left\{ \|\mathbf{f} - \mathbf{y}\|_2^2 + \gamma \mathbf{f}^{\mathrm{T}} \mathcal{L} \mathbf{f} \right\}$
  - optimal solution

$$f_*(i) = \sum_{\ell=0}^{N-1} \left[\frac{1}{1 + \gamma \lambda_\ell} \right] \hat{y}(\lambda_\ell) u_\ell(i) \quad \text{or} \quad \mathbf{f} = \hat{h}(\mathcal{L}) \mathbf{y} \text{ with } \hat{h}(\lambda) := \frac{1}{1 + \gamma \lambda}$$

Original Noisy Gaussian filtering Graph filtering

Filtering in the vertex domain

- Linear combination of values at neighbour vertices

$$f_{out}(i) = b_{i,i} f_{in}(i) + \sum_{j \in \mathcal{N}(i,K)} b_{i,j} f_{in}(j)$$

  - localized linear transform

- Example: polynomial filter as $\hat{h}(\lambda_\ell) = \sum_{k=0}^{K} a_k \lambda_\ell^k$

$$f_{out}(i) = \sum_{\ell=0}^{N-1} \hat{f}_{in}(\lambda_\ell) \hat{h}(\lambda_\ell) u_\ell(i)$$

$$= \sum_{j=1}^{N} f_{in}(j) \sum_{k=0}^{K} a_k \left(\mathcal{L}^k\right)_{i,j} \rightarrow b_{i,j} := \sum_{k=d_{\mathcal{G}}(i,j)}^{K} a_k \left(\mathcal{L}^k\right)_{i,j}$$

Localization of polynomials

- **Lemma** [Hammond:2011]**:** for any two vertices *i* and *j*, if the minimal hop distance $d_{\mathcal{G}}(i,j) > s$ then $\left(\mathcal{L}^s\right)_{i,j} = 0$

  - *Proof:* $\mathcal{L}_{i,j} = 0$ if *i* and *j* are not connected

    $\left(\mathcal{L}^s\right)_{i,j} = \sum \mathcal{L}_{i,k_1}\mathcal{L}_{k_1,k_2}\dots\mathcal{L}_{k_{s-1},j}$ over s-1 length sequences

    By contra. $\left(\mathcal{L}^s\right)_{i,j} \neq 0$ $\Longrightarrow$ at least one non-zero term in the sum

    $\Longrightarrow \exists$ a path of length $d_{\mathcal{G}}(i,j) \leq s$

- Kernels defined by smooth polynomial functions of the Laplacian are localised in the vertex domain

$$\hat{h}(\lambda_\ell) = \sum_{k=0}^{K} a_k \lambda_\ell^k \quad \text{and} \quad f_{in}(j) = \begin{cases} 1 & \text{if } j = n \\ 0 & \text{otherwise} \end{cases} \quad \Longrightarrow \quad f_{out}(i) = \sum_{k=0}^{K} a_k \left(\mathcal{L}^k\right)_{i,n}$$

localized within K hops of *n* !

Convolution

- The classical convolution does not generalise to the graph settings
  - $h(t-\tau)$ does not have any equivalent on graphs

$$f_{out}(t) = \int_{\mathbb{R}} f_{in}(\tau)h(t-\tau)d\tau =: (f_{in} * h)(t)$$

- Instead, it can be defined by multiplication in the graph spectral domain

$$(f * h)(i) := \sum_{\ell=0}^{N-1} \hat{f}(\lambda_\ell)\hat{h}(\lambda_\ell)u_\ell(i)$$

Translation on graphs

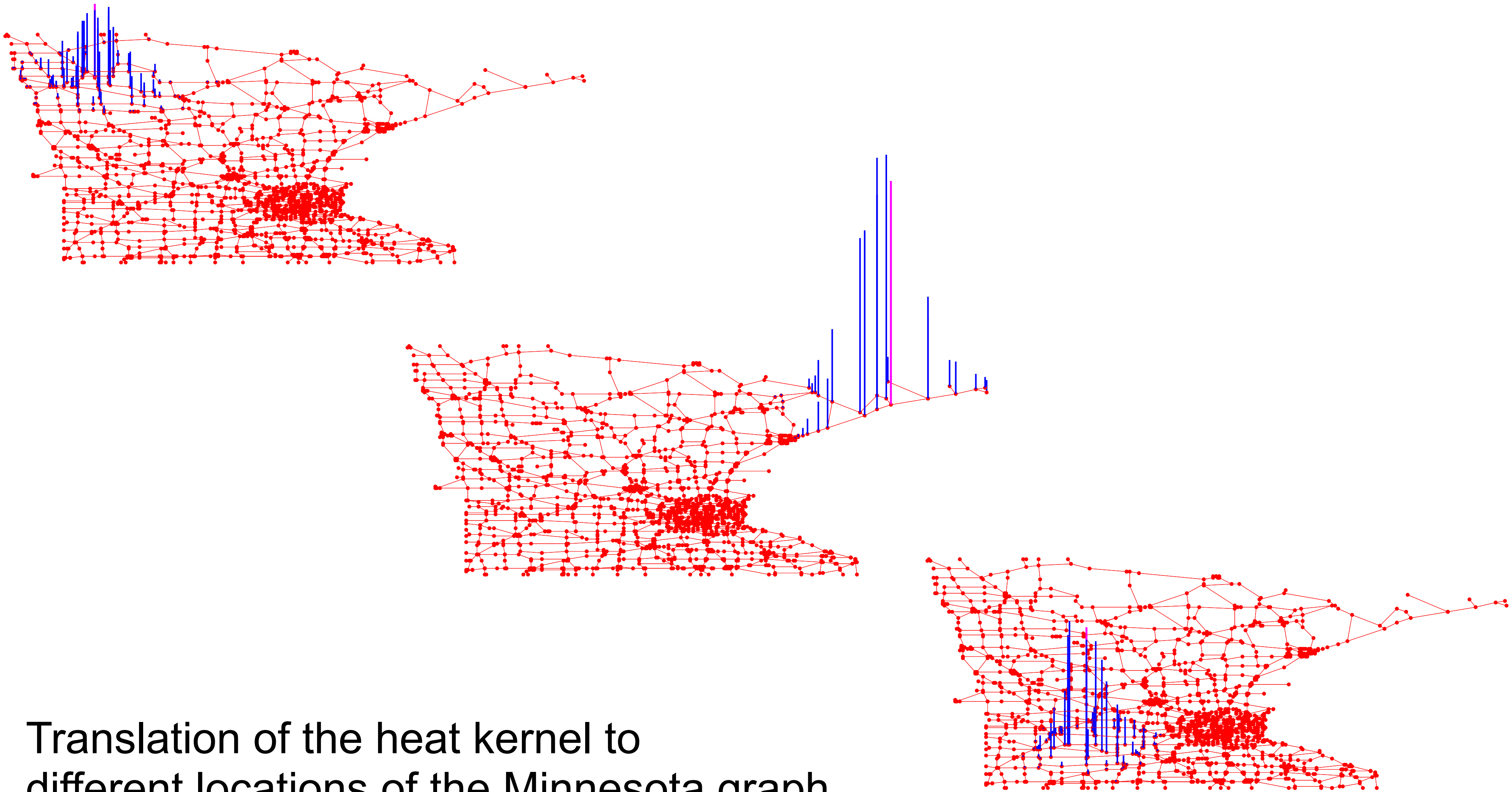
- The classical translation $(T_u f)(t) := f(t-u)$ does not generalise to non-regular graphs
- A generalized translation operator on graphs can still be defined as

$$T_n : \mathbb{R}^N \to \mathbb{R}^N$$

$$(T_n g)(i) := \sqrt{N}(g * \delta_n)(i) = \sqrt{N} \sum_{\ell=0}^{N-1} \hat{g}(\lambda_\ell) u_\ell^*(n) u_\ell(i)$$

$$\delta_n(i) = \begin{cases} 1 & \text{if } i = n \\ 0 & \text{otherwise} \end{cases}$$

Translation example



Translation of the heat kernel to
different locations of the Minnesota graph

Transforms on graphs

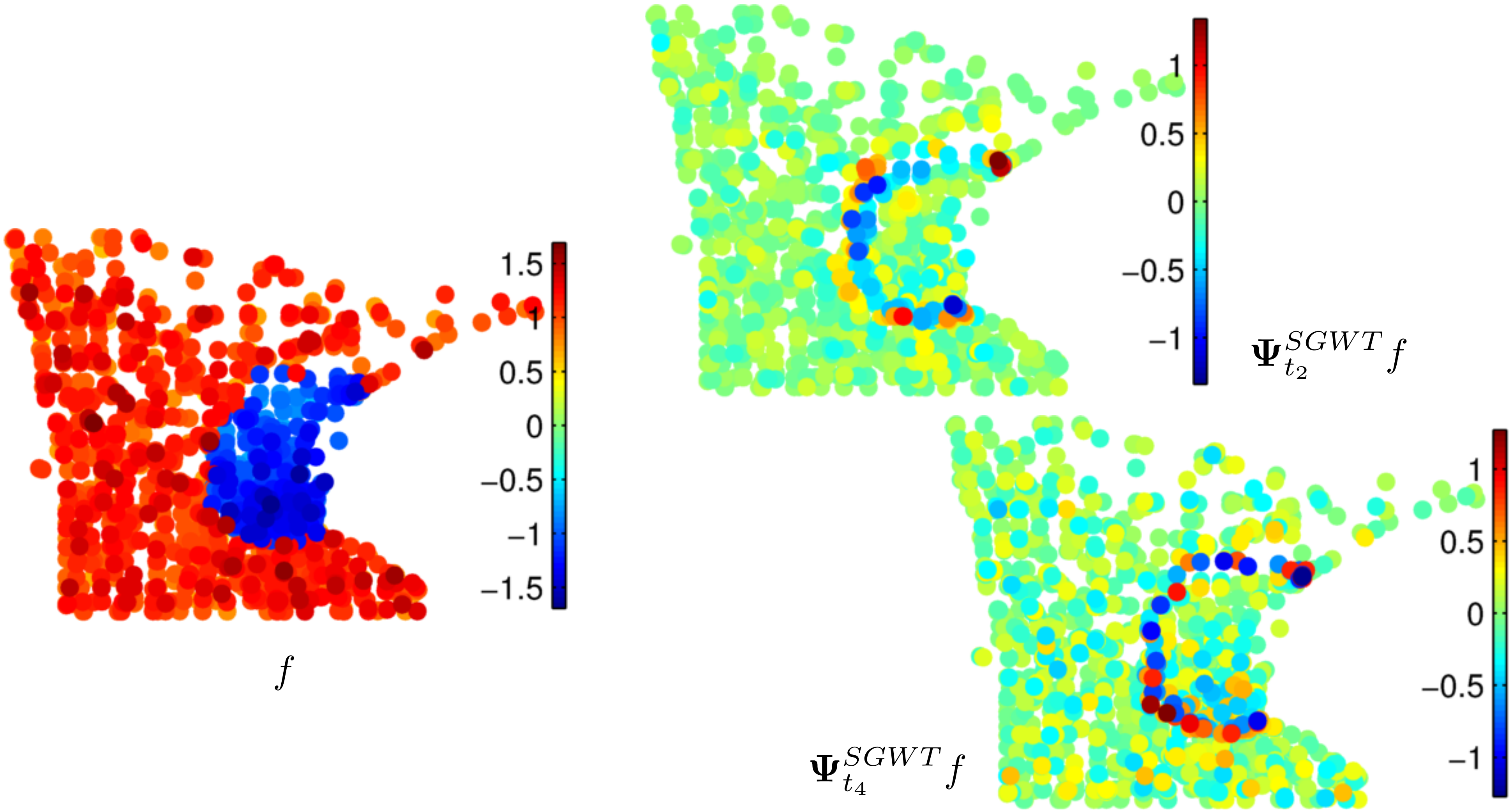
- Localized transforms are ideal to analyse graph signals
  - analysis properties and scalable implementations
- Wavelet transforms are particularly interesting
  - localization in both the vertex and spectral domains
  - different designs in the vertex or the spectral domain [Shuman:2013]
  - *Example*: Spectral Graph Wavelets [Hammond:2011]

$$\boldsymbol{\Psi}^{SGWT} : \mathbb{R}^N \to \mathbb{R}^{N(K+1)} \qquad \boldsymbol{\Psi}^{SGWT} = [\boldsymbol{\Psi}^{SGWT}_{scal}; \boldsymbol{\Psi}^{SGWT}_{t_1}; \ldots; \boldsymbol{\Psi}^{SGWT}_{t_K}]$$

- Dilations and translations of a band-pass kernel $\psi^{SGWT}_{t_k,i} := T_i \mathcal{D}_{t_k} \mathbf{g} = \widehat{\mathcal{D}_{t_k} g}(\mathcal{L}) \boldsymbol{\delta}_i$
- Translation of a low-pass kernel $\psi^{SGWT}_{scal,i} := T_i \mathbf{h} = \hat{h}(\mathcal{L}) \boldsymbol{\delta}_i$

- Such transforms do not explicitly adapt to the data :(

SGWT illustration



[Shuman:2013]

Another SGWT illustration

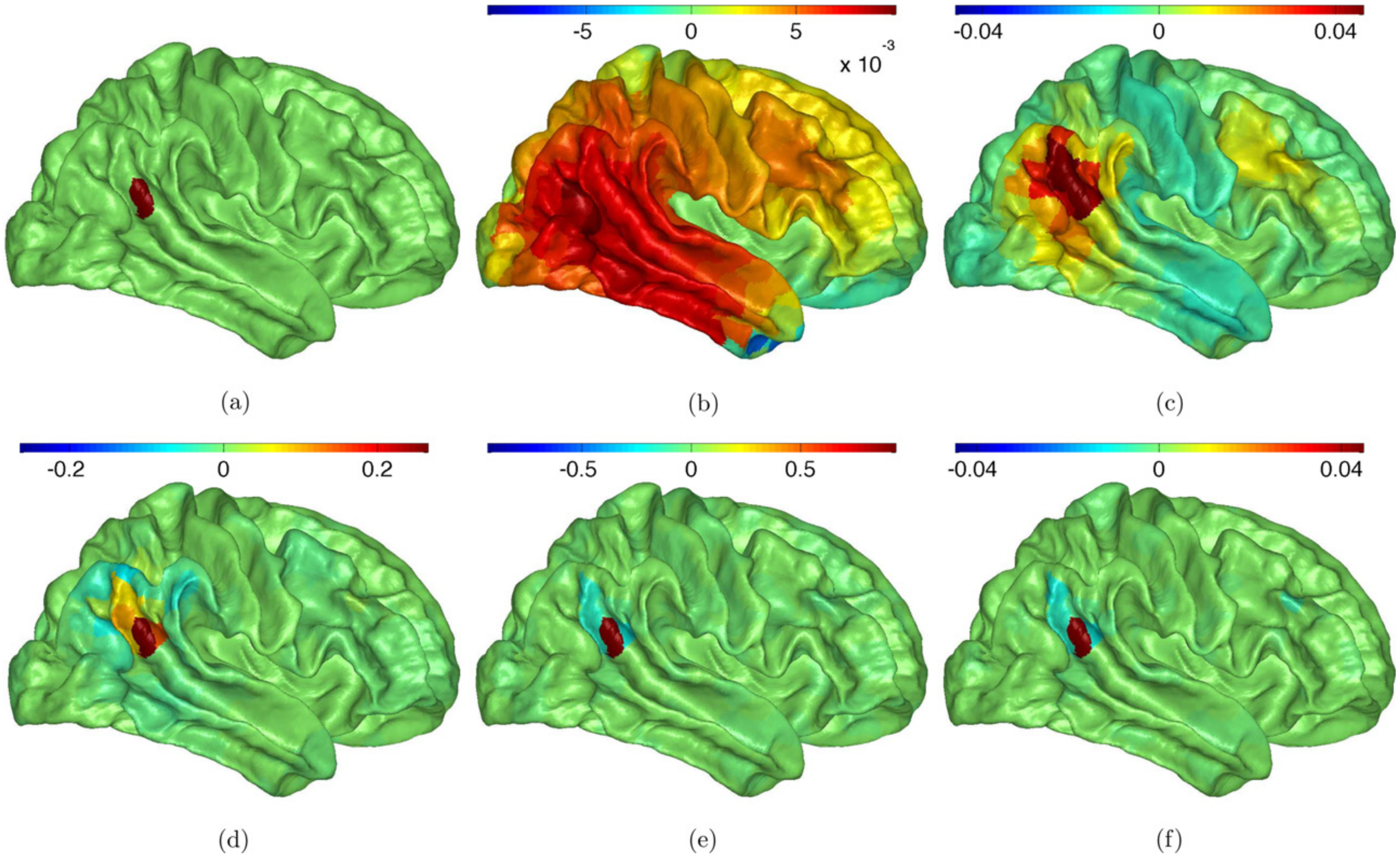
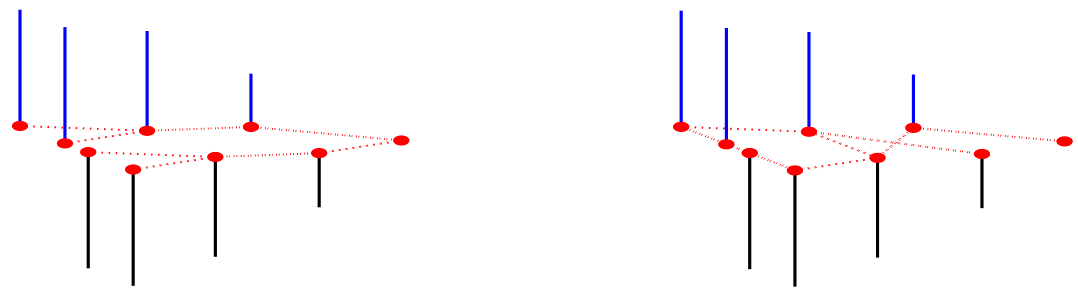


Fig. 5. Spectral graph wavelets on cerebral cortex, with $K = 50$, $J = 4$ scales. (a) ROI at which wavelets are centered, (b) scaling function, (c)–(f) wavelets, scales 1–4.

[Hammond:2011]

Better adaptation to data?

- The representation can be adapted to the data by numerical optimisation
- Dictionary Learning could be performed naively on graph signals represented as vectors
  - K-SVD, Method of Optimal Directions (MOD), etc
  - Agnostic to the graph structure :(
    - Permutation of indexes changes the dictionary
    - Different graph signals with the same representation



- Costly, highly non-structured representations :(

Bridging the gap…

- Sparse graph signal representation

$$\mathbf{y} = \mathbf{\Phi x} \quad \text{s.t.} \quad \|\mathbf{x}\|_0 \leq T_0$$

structure-based methods

$\mathbf{\Phi}$ defined
via mathematical structures
(*transforms*):
Fourier, wavelets...

not data-specific!

numerical methods

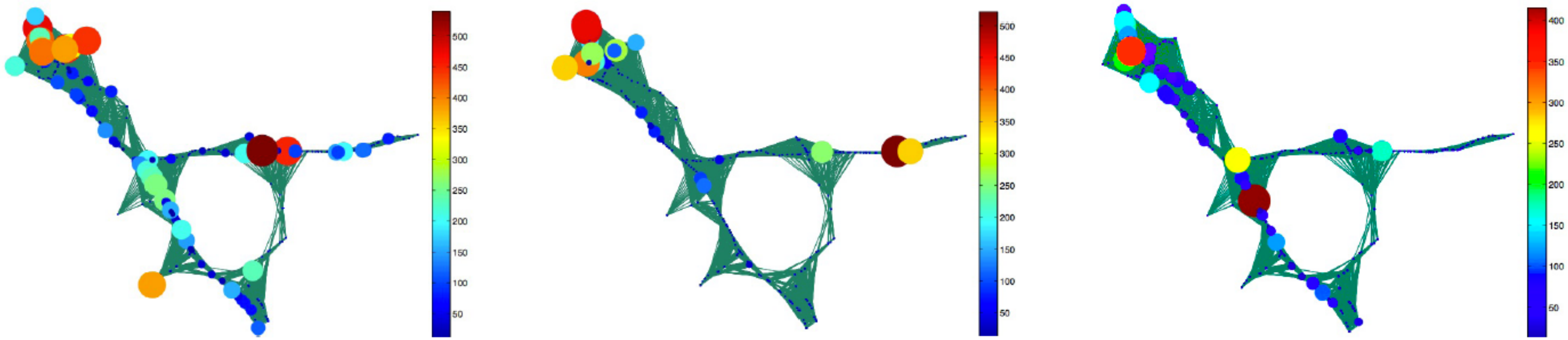
$\mathbf{\Phi}$ learned
via training samples
(*dictionary learning*):
sparse coding, MOD, K-SVD...

not graph-specific!

- We want to have an efficient structured representation $\mathbf{\Phi}$ that is adapted to data: *graph spectral dictionaries*

Dictionary for Graph Signals

- Our objective: meaningful graph signal representations that
  - ✓ reveal relevant structural properties of the graph signals/extract important features on graphs
  - ✓ sparsely represent different classes of signals on graphs



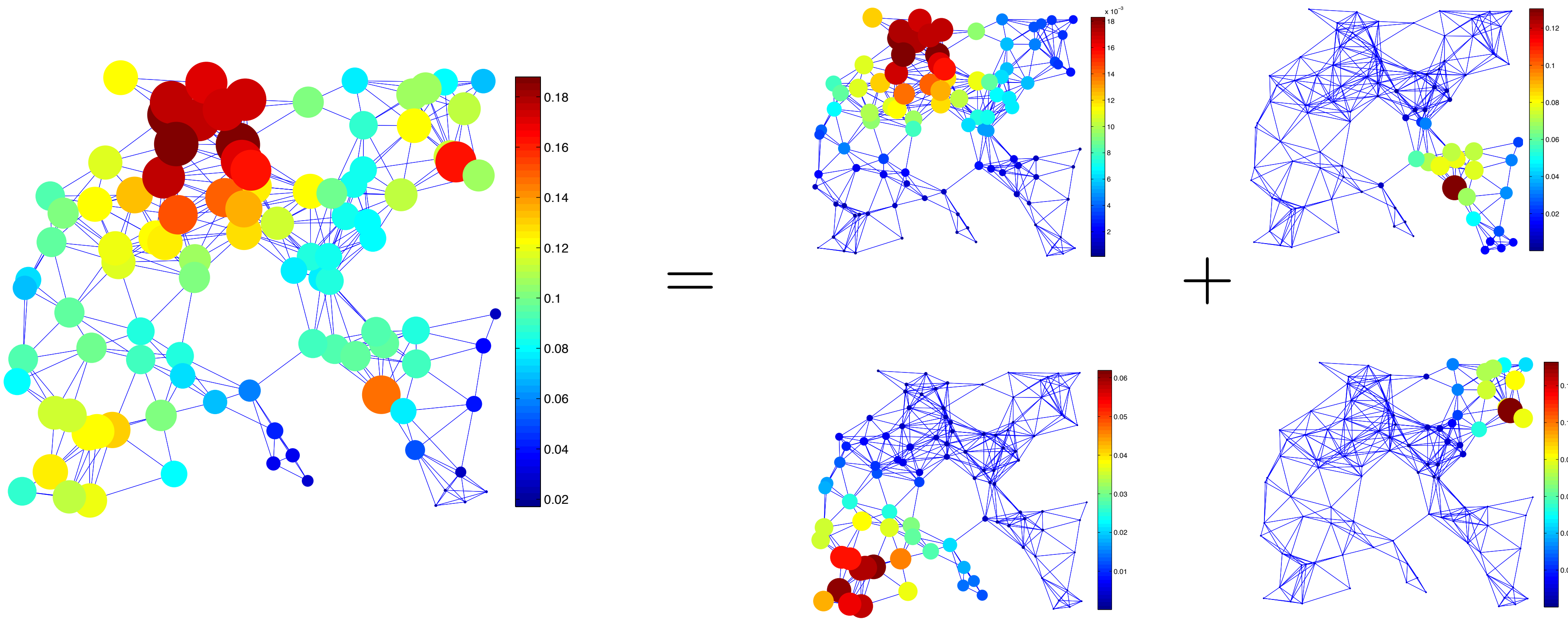
(a) Day 1 (b) Day 2 (c) Day 3

How can we define atoms on graphs?

Sparse signal model

- Graph signals can be approximated by a small number of localized components
  - e.g., multiple processes started at different vertices

$e^{-\tau_1 \mathcal{L}} \delta_i, \tau_1 = 15$ $\quad$ $e^{-\tau_2 \mathcal{L}} \delta_i, \tau_2 = 3$



Parametric graph atoms

- A set of generating kernels $\{\widehat{g_s}(\cdot)\}_{s=1,2,\ldots,S}$ represent the spectral characteristics of the signals
- The kernels are chosen to be smooth polynomial of degree K in order to form localized graph features

$$\hat{g}(\lambda_\ell) = \sum_{k=0}^{K} \alpha_k \lambda_\ell^k, \quad \ell = 0, \ldots, N-1$$

- A graph atom is the translation of the kernel to vertex *n*

$$T_n g = \sqrt{N}(g * \delta_n) = \sqrt{N} \sum_{\ell=0}^{N-1} \sum_{k=0}^{K} \alpha_k \lambda_\ell^k \chi_\ell^*(n) \chi_\ell = \sqrt{N} \sum_{k=0}^{K} \alpha_k (\mathcal{L}^k)_n$$

$\mathcal{L}$: normalized Laplacian, χ_ℓ : eigenvector

Dictionary Structure

- A parametric graph dictionary $\mathcal{D} = [\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_S]$ is a concatenation of S subdictionaries

- Each subdictionary is built on a specific kernel

$$\mathcal{D}_s = \widehat{g}_s(\mathcal{L}) = \chi \left(\sum_{k=0}^{K} \alpha_{sk} \Lambda^k \right) \chi^T = \sum_{k=0}^{K} \alpha_{sk} \mathcal{L}^k$$

  - Each atom (column of $\mathcal{D}_s$ corresponds to a K-hop localized pattern centered on a node of the graph, i.e.,

$$\frac{1}{\sqrt{N}} T_n g_s$$

Dictionary design constraints

- The kernels have to be nonnegative and bounded

$$0 \leq \widehat{g}_s(\lambda) \leq c, \ \forall \lambda \in [0, \lambda_{\max}]$$

$$0 \preceq \mathcal{D}_s \preceq cI, \quad \forall s \in \{1, 2, ..., S\},$$

  - Each subdictionary is positive semi-definite with max eigenvalue bounded by *c*

- The kernels should cover the full spectrum

$$c - \epsilon_1 \leq \sum_{s=1}^{S} \widehat{g}_s(\lambda) \leq c + \epsilon_2, \ \text{for all } \lambda \in [0, \lambda_{\max}]$$

$$(c - \epsilon_1)I \preceq \sum_{s=1}^{S} \mathcal{D}_s \preceq (c + \epsilon_2)I$$

Frame bounds

- With

$$\begin{cases} \mathcal{D}_s = \sum_{k=0}^{K} \alpha_{sk} \mathcal{L}^k \\ 0 \leq \widehat{g_s}(\lambda) \leq c, \ \forall \lambda \in [0, \lambda_{\max}] \\ c - \epsilon_1 \leq \sum_{s=1}^{S} \widehat{g_s}(\lambda) \leq c + \epsilon_2, \ \text{for all } \lambda \in [0, \lambda_{\max}] \end{cases}$$

the set of atoms $\{d_{s,n}\}_{s=1,2,\ldots,S,n=1,2,\ldots,N}$ form a frame

$$\frac{(c-\epsilon_1)^2}{S} \|y\|_2^2 \leq \sum_{n=1}^{N} \sum_{s=1}^{S} |\langle y, d_{s,n} \rangle|^2 \leq (c+\epsilon_2)^2 \|y\|_2^2 \qquad \forall y \in \mathbb{R}^N$$

Proof

By generalisation of the Theorem 5.6 in [Hammond:2011]

$$\sum_{n=1}^{N}\sum_{s=1}^{S}|\langle y, d_{s,n}\rangle|^2 = \sum_{\ell=0}^{N-1}|\hat{y}(\lambda_\ell)|^2\sum_{s=1}^{S}|\widehat{g_s}(\lambda_\ell)|^2, \quad \forall\lambda\in\sigma(\mathcal{L}). \tag{1}$$

From the constraints on the spectrum of kernels $\{\widehat{g_s}(\cdot)\}_{s=1,2,\dots,S}$ we have

$$\sum_{s=1}^{S}|\widehat{g_s}(\lambda_\ell)|^2 \le \left(\sum_{s=1}^{S}\widehat{g_s}(\lambda_\ell)\right)^2 \le (c+\epsilon_2)^2, \quad \forall\lambda\in\sigma(\mathcal{L}). \tag{2}$$

Moreover, from the left side of the second design constraint and the Cauchy-Schwarz inequality, we have

$$\frac{(c-\epsilon_1)^2}{S} \le \frac{\left(\sum_{s=1}^{S}\widehat{g_s}(\lambda_\ell)\right)^2}{S} \le \sum_{s=1}^{S}|\widehat{g_s}(\lambda_\ell)|^2, \quad \forall\lambda\in\sigma(\mathcal{L}). \tag{3}$$

Combining (1), (2) and (3) yields the desired result.

Dictionary Learning Problem

- Learning consists in computing $\{\alpha_{sk}\}_{s=1,2,...,S;\ k=1,2,...,K}$
- Given a set of training signals $Y = [y_1, y_2, ..., y_M] \in \mathbb{R}^{N \times M}$ on the graph $\mathcal{G}$, solve

$$\begin{aligned} \underset{\alpha \in \mathbb{R}^{(K+1)S},\ X \in \mathbb{R}^{SN \times M}}{\operatorname{argmin}} \quad & \{ ||Y - \mathcal{D}X||_F^2 + \mu \|\alpha\|_2^2 \} \\ \text{subject to} \quad & \|x_m\|_0 \leq T_0, \quad \forall m \in \{1, ..., M\}, \\ & \mathcal{D}_s = \sum_{k=0}^{K} \alpha_{sk} \mathcal{L}^k, \quad \forall s \in \{1, 2, ..., S\} \\ & 0 \preceq \mathcal{D}_s \preceq c, \quad \forall s \in \{1, 2, ..., S\} \\ & (c - \epsilon_1) I \preceq \sum_{s=1}^{S} \mathcal{D}_s \preceq (c + \epsilon_2) I, \end{aligned}$$

The spectral constraints guarantee that:
1. The learned kernels cover the whole spectrum
2. The dictionary is a frame

Alternating optimisation

Algorithm 1 Parametric Dictionary Learning on Graphs

Input: Signal set Y, initial dictionary $\mathcal{D}^{(0)}$, target signal sparsity T_0, polynomial degree K, number of subdictionaries S, number of iterations $iter$
Output: Sparse signal representations X, polynomial coefficients α
Initialization: $\mathcal{D} = \mathcal{D}^{(0)}$
for $i = 1, 2, ..., iter$ **do:**
    Sparse Approximation Step:
        (a) Scale each atom in $\mathcal{D}$ to a unit norm
        (b) Update X using Sparse Coding
        (c) Rescale X, $\mathcal{D}$ to recover the polynomial structure
    Dictionary Update Step:
        Compute the polynomial coefficients α and update the dictionary
end for

Sparse Coding Step

- The dictionary (α) is fixed
- The sparse coding coefficients are computed with

$$\underset{X}{\operatorname{argmin}} ||Y - \mathcal{D}X||_F^2 \text{ subject to } \|x_m\|_0 \leq T_0$$

$$\forall m \in \{1, ..., M\}$$

- this can be solved by greedy algorithmms, like OMP
- it can also be solved by convex relaxation using iterative soft thresholding, for example

Dictionary Update Step

- The coefficients *X* are fixed, the dictionary is updated with

$$\underset{\alpha \in \mathbb{R}^{(K+1)S}}{\operatorname{argmin}} \left\{ ||Y - \mathcal{D}X||_F^2 + \mu \|\alpha\|_2^2 \right\}$$

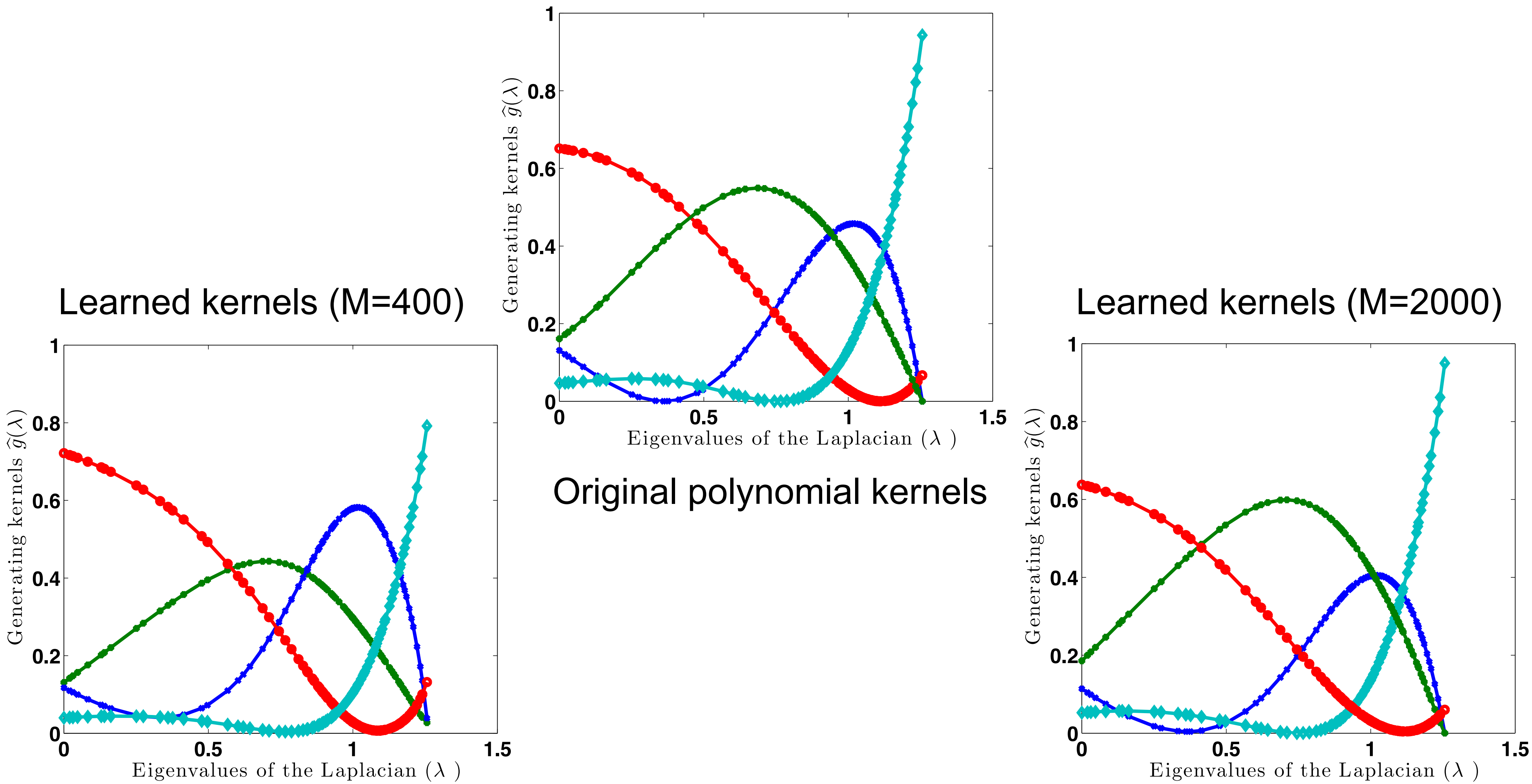
$$\text{subject to} \quad \mathcal{D}_s = \sum_{k=0}^{K} \alpha_{sk} \mathcal{L}^k, \quad \forall s \in \{1, 2, ..., S\}$$

$$0 \preceq \mathcal{D}_s \preceq cI, \quad \forall s \in \{1, 2, ..., S\}$$

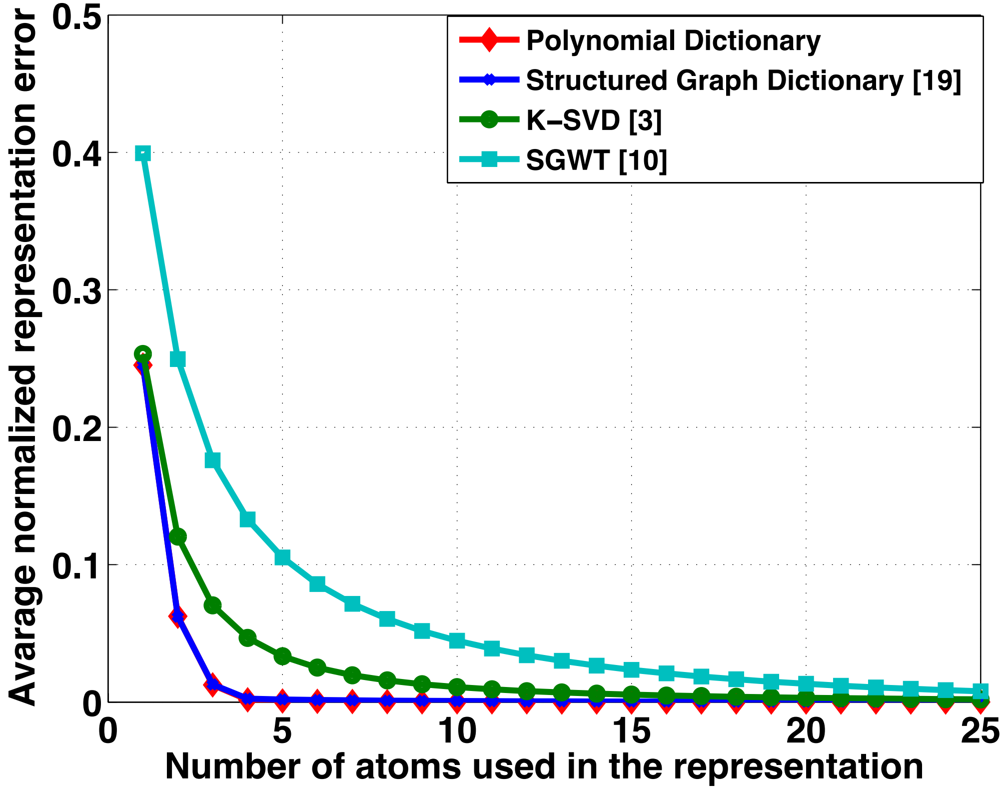
$$(c - \epsilon_1) I \preceq \sum_{s=1}^{S} \mathcal{D}_s \preceq (c + \epsilon_2) I.$$

- quadratic function with affine constraints, solved by interior point methods or ADMM

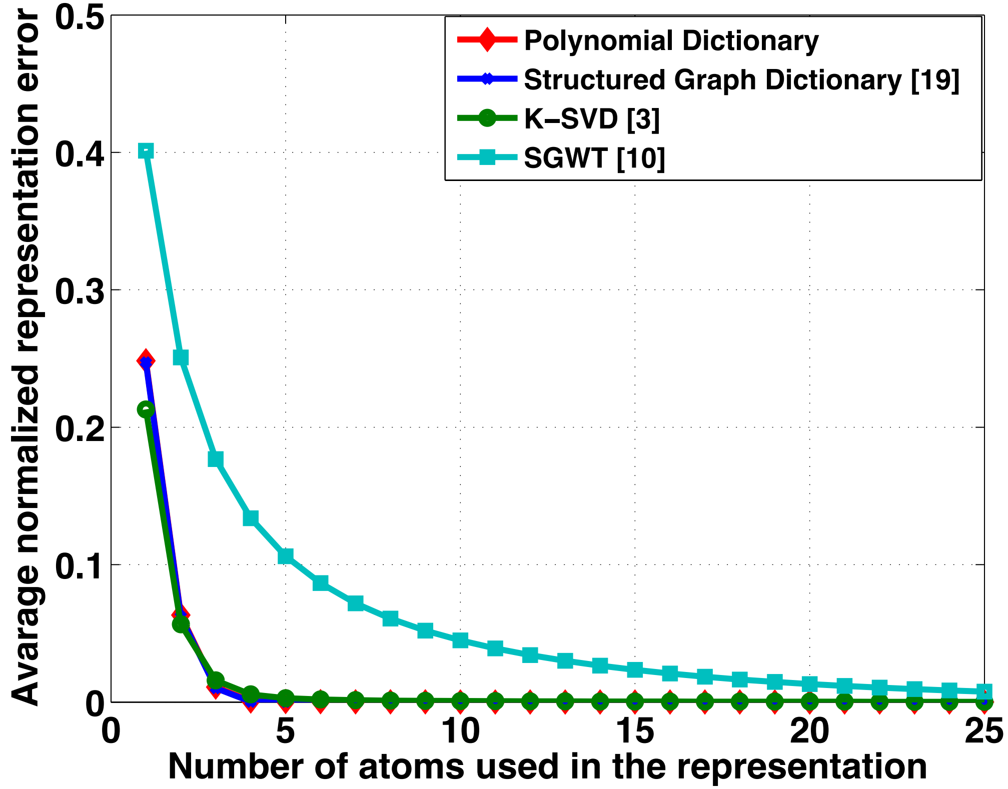
Recovery on synthetic data



Approximation on synthetic data

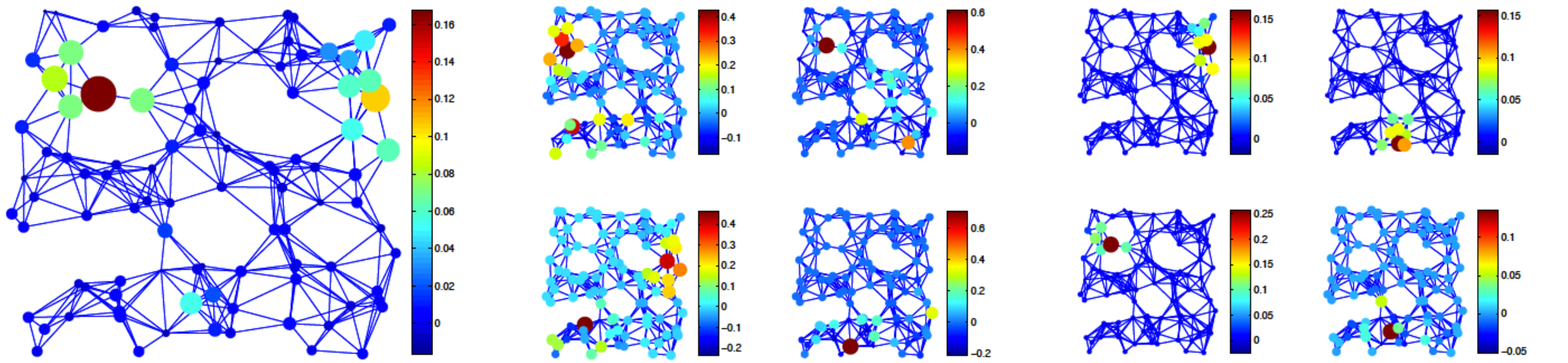


M=400



M=2000

Examples of atoms



(a) Graph Signal

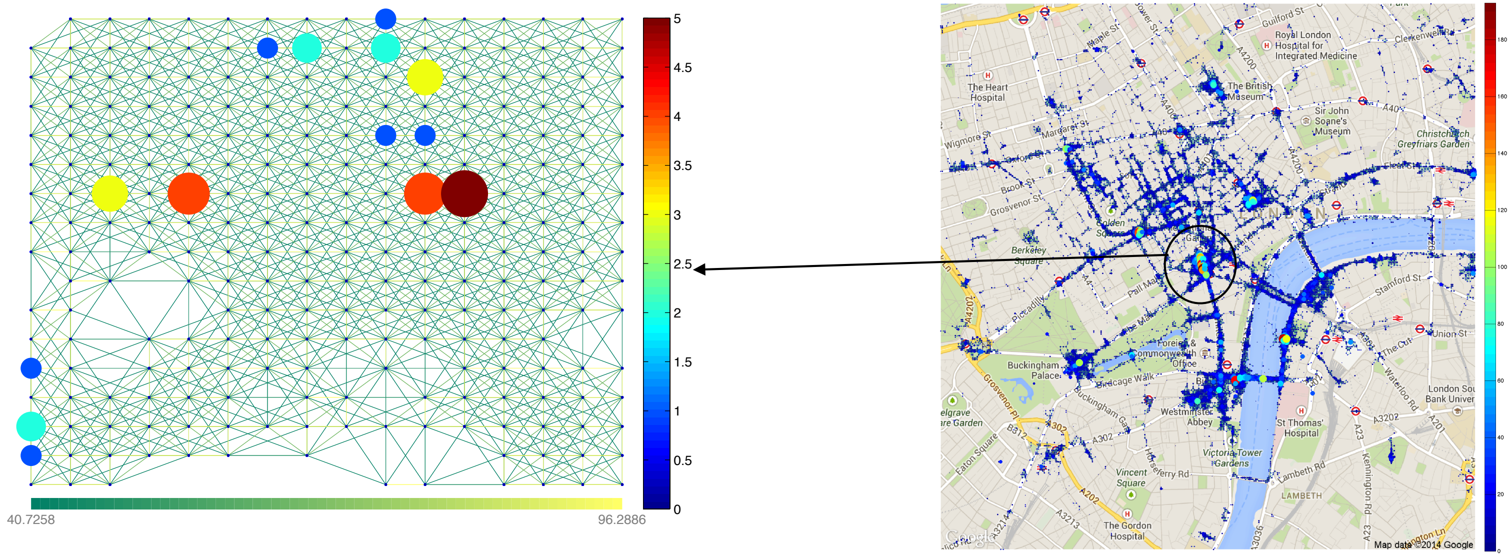
(b) Atomic decomposition with OMP in the K-SVD dictionary

(c) Atomic decomposition with OMP in the Polynomial dictionary

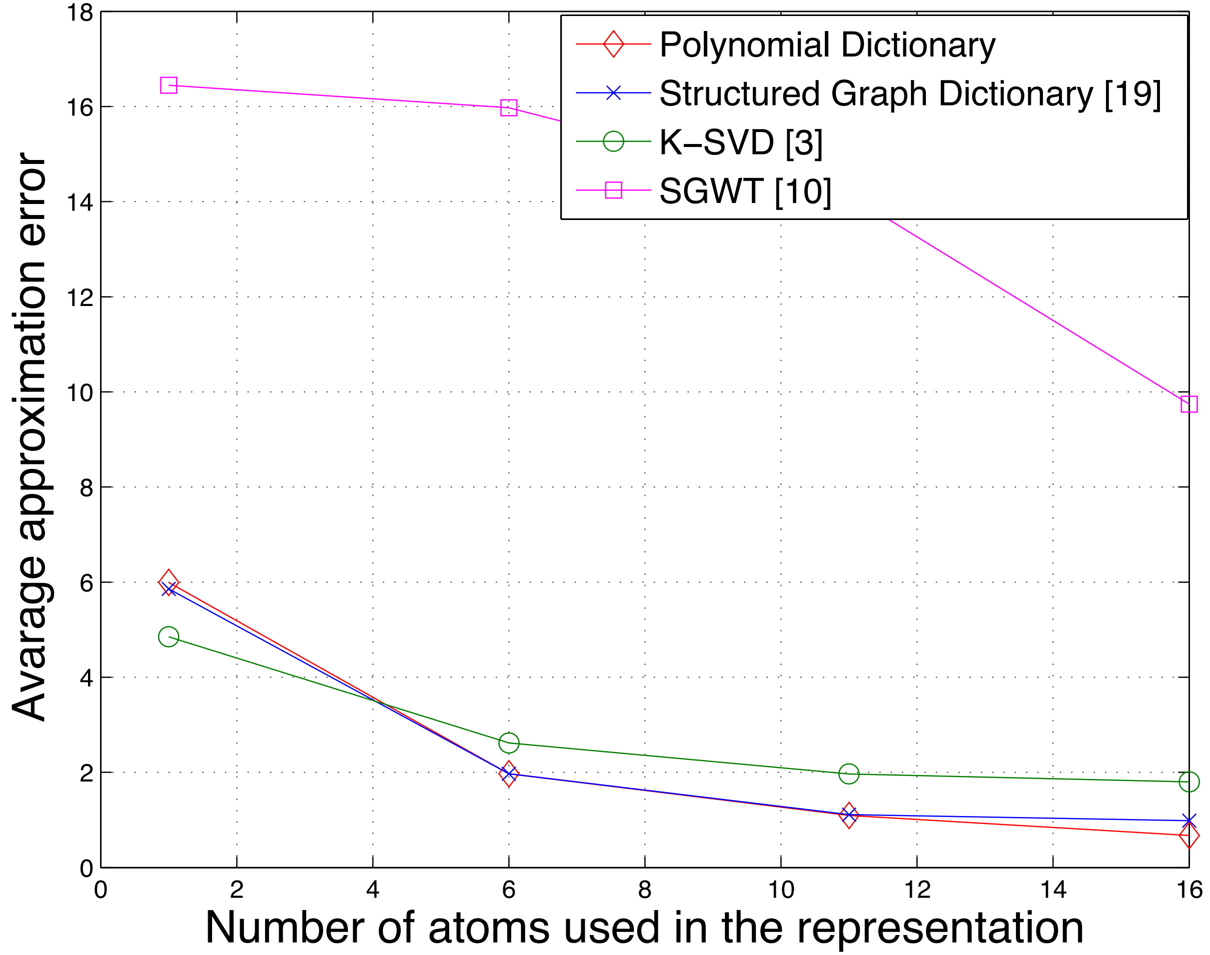
The K-SVD atoms have global support while the polynomial dictionary atoms are well localized on the graph

Flickr dataset

- Nodes: 245 vertices around Trafalgar Square (London), each representing a geographical area 10x10m^2
- Assign edges when distance < 30m
- Graph Signals: Daily number of distinct users that took photos between Jan. 2010 and June 2012

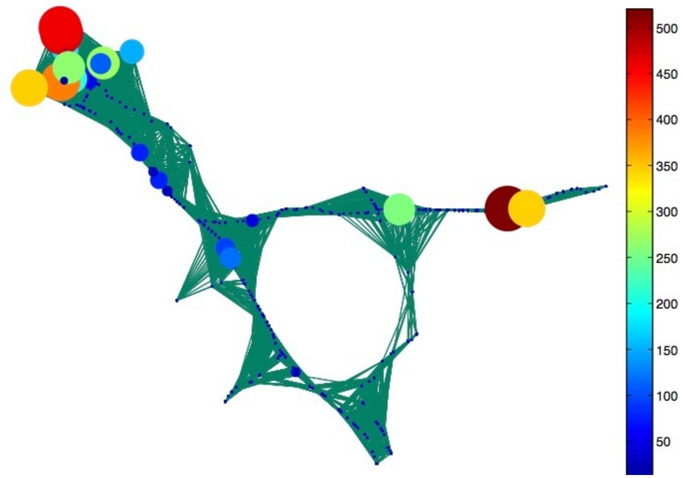


Flickr signal approximation

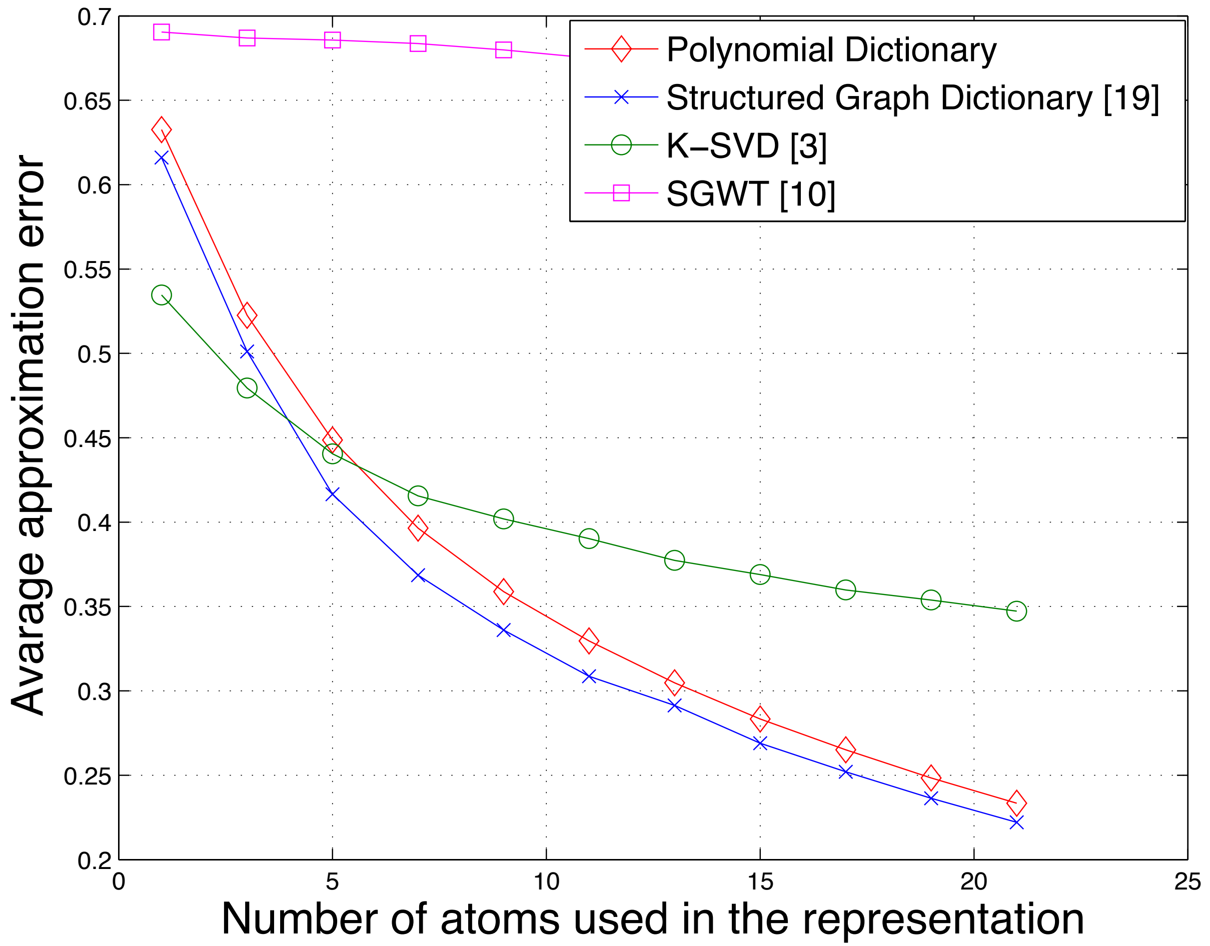


Traffic dataset

- Nodes: 439 detector stations in Alameda County, CA
- Assign edge when distance < 13km
- Graph Signals: Daily number of bottlenecks (in minutes) between Jan. 2007 to May. 2013

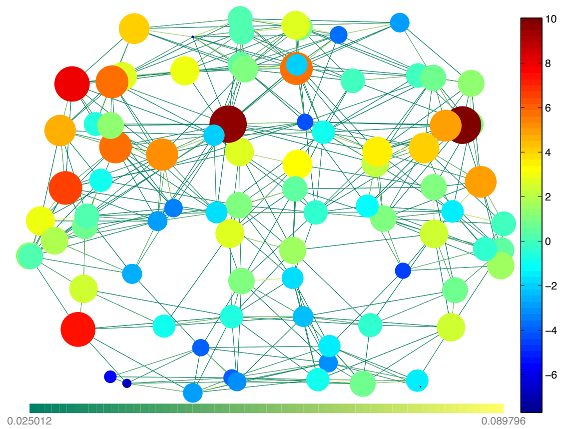


Traffic signal approximation

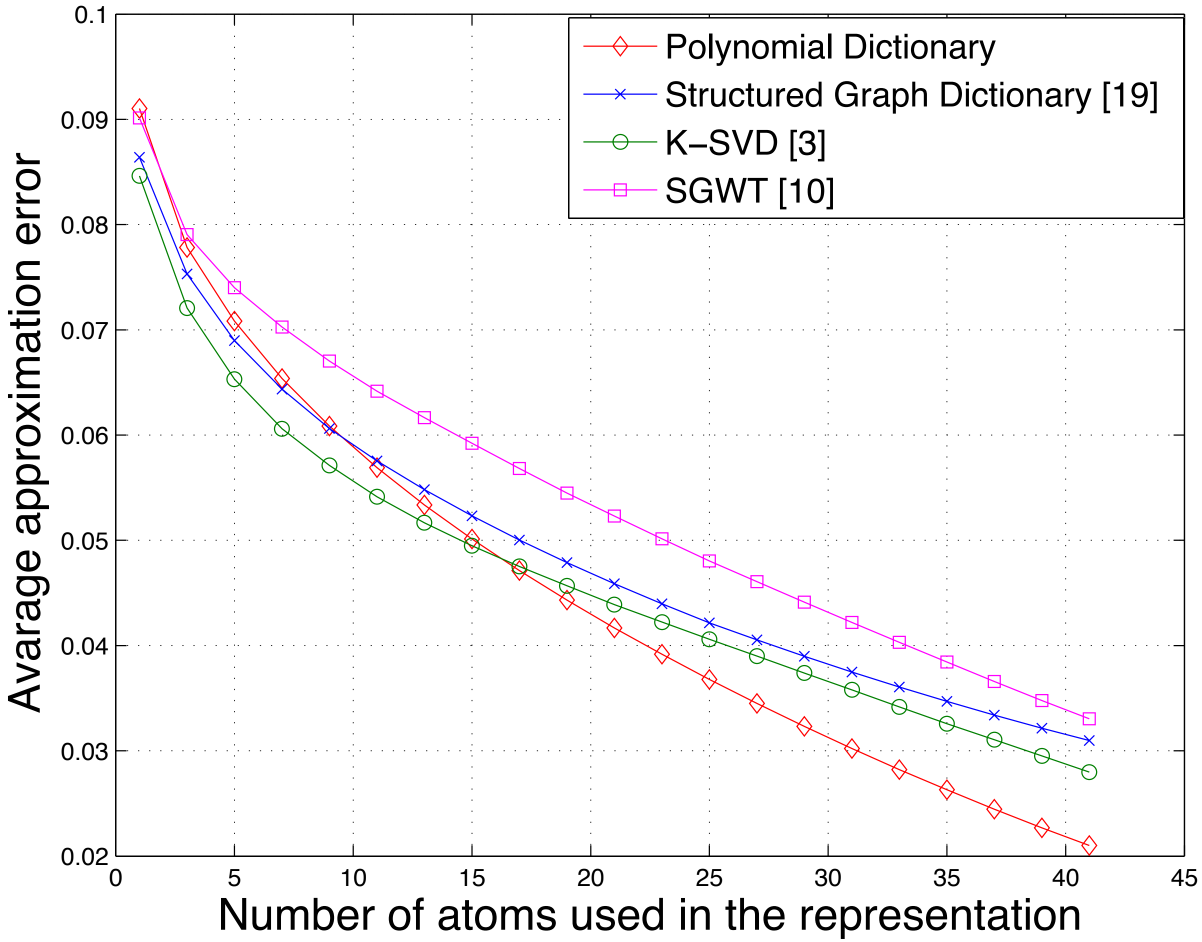


Brain dataset

- Nodes: 90 brain regions of contiguous voxels
- Edges assigned if anatomical distance < 40 mm
- Graph Signals: fMRI signals acquired on five subjects, in different states - 1290 signals per subject

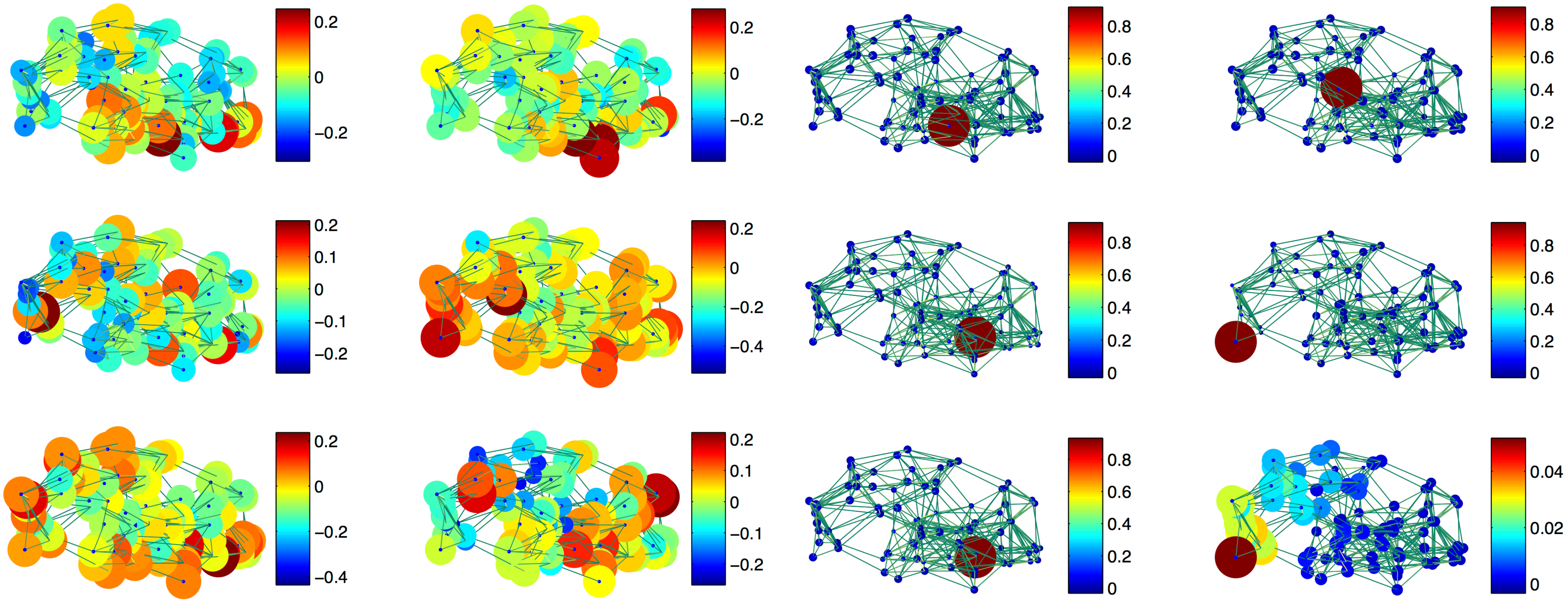


Brain signal approximation



Examples of Learned Atoms

- Most common atoms in OMP expansions



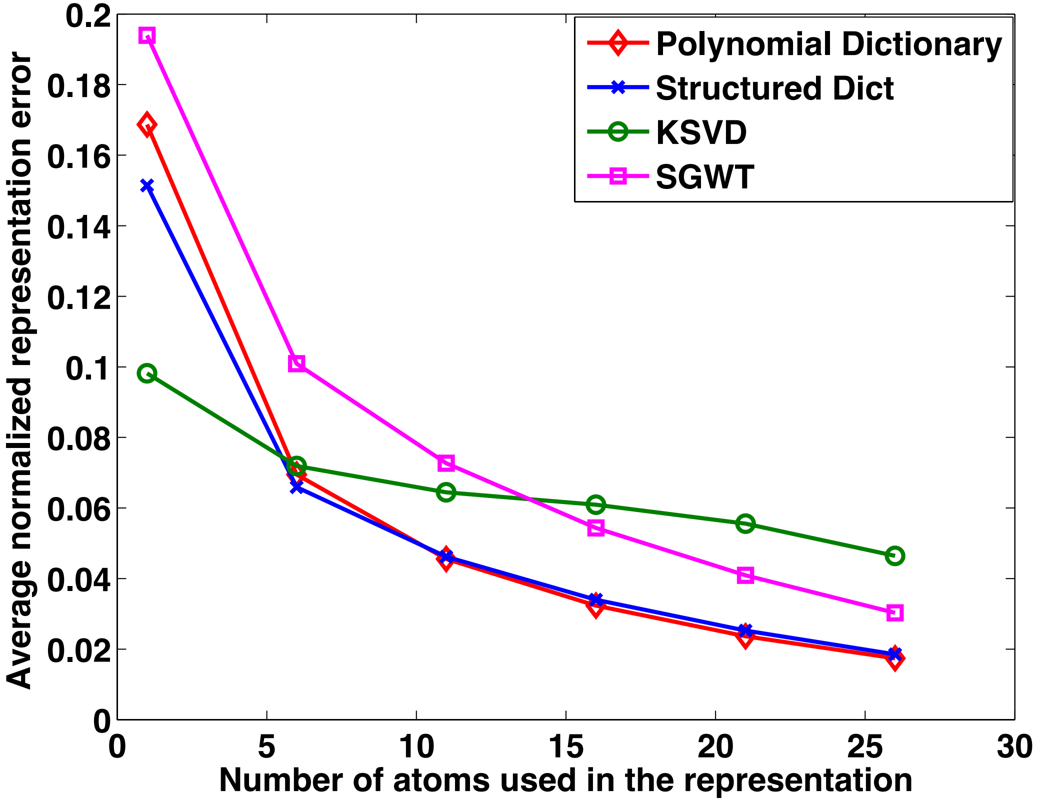
K-SVD Dictionary

Polynomial Graph Dictionary

Twitter dataset

- Graph: a social network of 63 Twitter users

- Training & Testing signals: 4032 & 4032 signals with number of tweets that each user has posted during several time intervals



Benefits of the structure

- The dictionary is easy to describe (e.g., store, or transmit)
  - it has only $(K+1)S$ parameters

- Efficient implementation, esp. when the graph is sparse
  - Both forward and adjoint operators can be efficiently applied
  - Both operators are the main components of many sparsity-based applications

$\mathcal{D}^T y = \sum_{s=1}^{S} \sum_{k=0}^{K} \alpha_{sk} \mathcal{L}^k y$ is $O(K|\mathcal{E}| + NSK)$ since $\{\mathcal{L}^k y\}_{k=0,2,\ldots,K}$ is $O(K|\mathcal{E}|)$

$\mathcal{D}\mathcal{D}^T y = \sum_{s=1}^{S} \widehat{g_s}^2(\mathcal{L}) y$ similar, with a polynomial of degree $K' = 2K$

Example: Iterative soft thresholding

- Lasso regularisation problem

$$x^* = \min_x \|y - \mathcal{D}x\|_2^2 + \kappa \; \|x\|_1$$

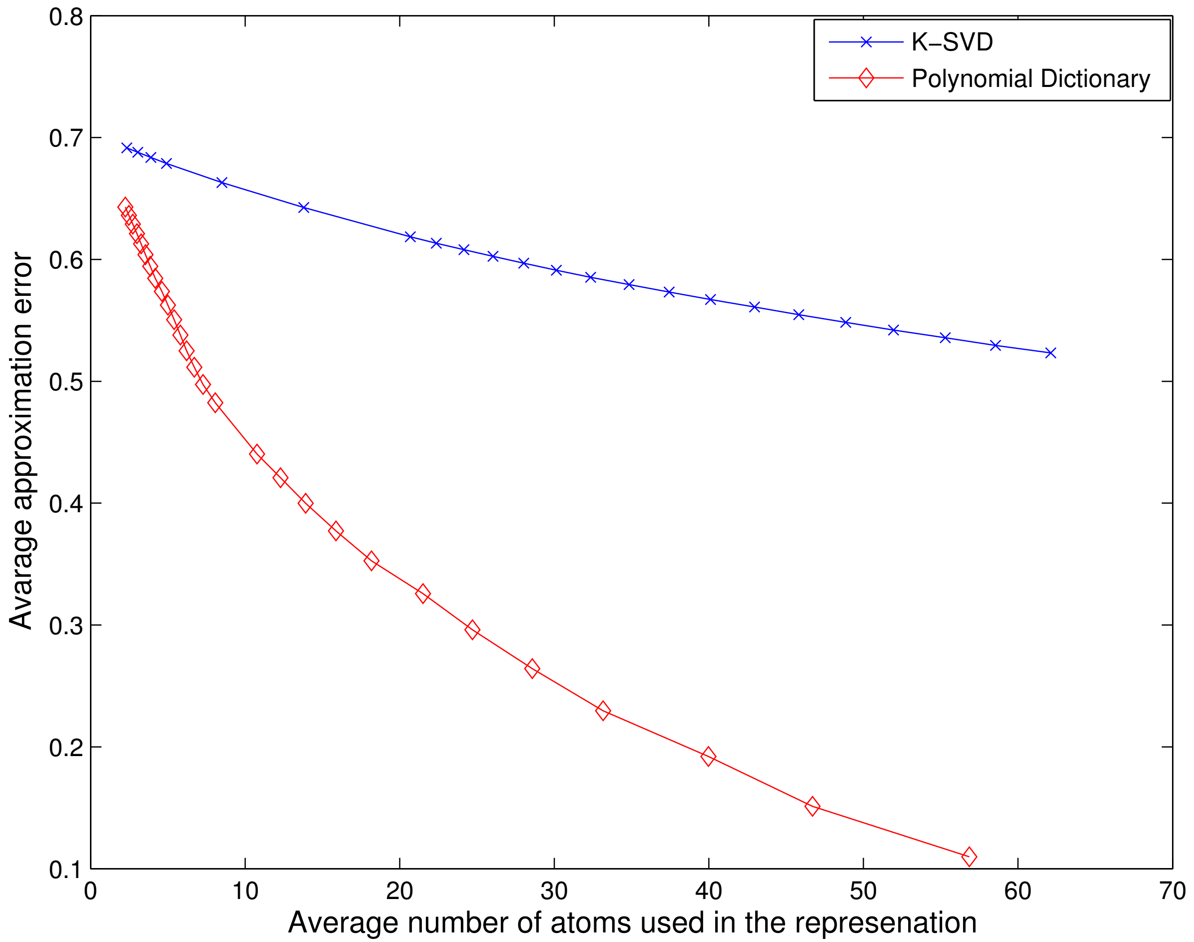
- It can be solved by iterative soft thresholding with

$$x^t = \mathcal{S}_{\kappa\tau}\left(x^{(t-1)} + 2\tau\mathcal{D}^T \; (y - \mathcal{D}x^{(t-1)})\right), \; t = 1, 2, \ldots$$

$$\mathcal{S}_{\kappa\tau} = \begin{cases} 0 & \text{if } |z| \leq \mu\tau \\ z - \operatorname{sgn}(z)\kappa\tau & \text{otherwise} \end{cases}$$

  - both dictionary-based operators are ‘easy’ to compute

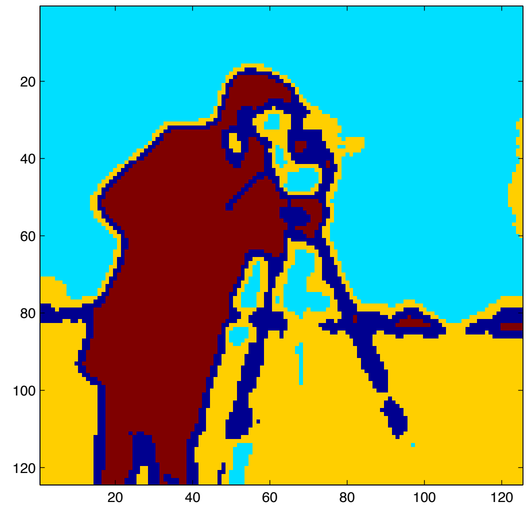
Illustrative Lasso performance



Iterative soft thresholding on traffic bottleneck signals

Applications of graph dictionaries

- Graph dictionaries apply to many sparse problems
  - sparsity prior on graphs
  - helpful when smooth priors are insufficient
- Graph dictionaries also define features on graphs
  - learning or clustering applications
- By construction, spectral graph dictionaries lead to effective implementations
  - distributed processing applications in networks



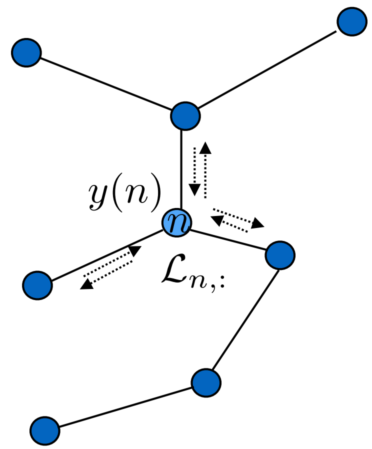


Image Segmentation Example

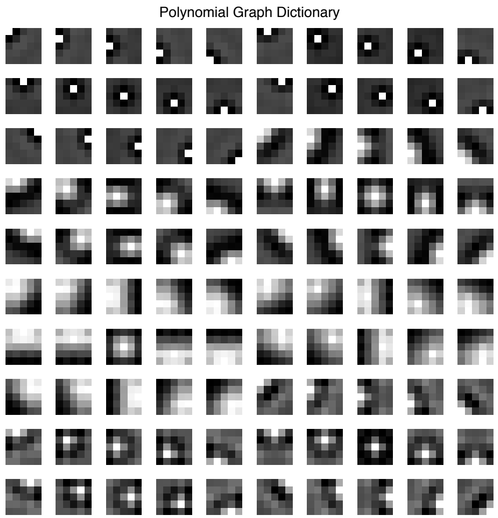
- Dictionary construction
  - For each pixel (node), build a 5x5 patch
    - each pixel connected to its horizontal and vertical neighbors
    - graphs signal is the pixel luminance value
  - Learn a dictionary from patch signals with $S = 4, K = 15$

- Segmentation
  - Process each signal with the learned filters i.e.,

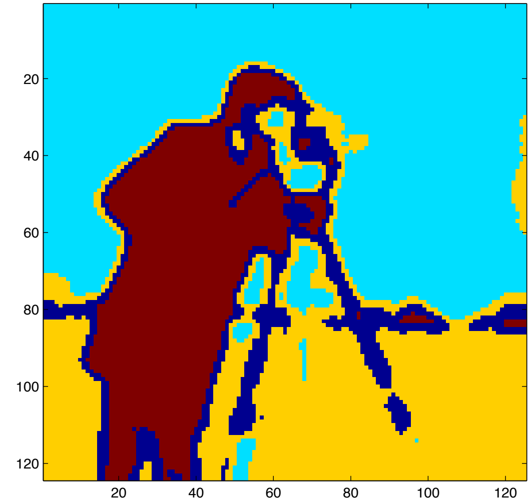
$$\mathcal{D}_s^T y_j = \sum_{\ell=0}^{N-1} \widehat{y_j}(\lambda_\ell)\widehat{g_s}(\lambda_\ell)\chi_\ell$$

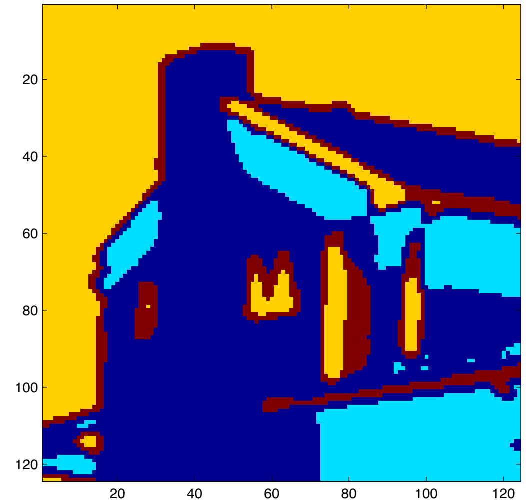
  - Node feature: mean and variance of the filtered signals
  - Clustering: K-means on the feature vectors

Clustering results



Atoms learned on patches

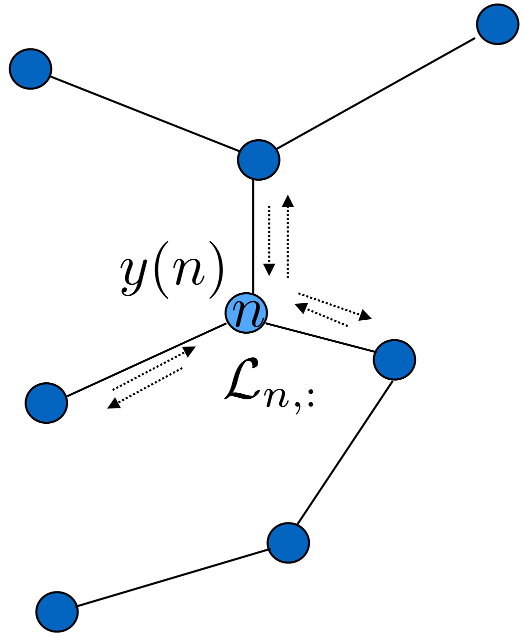




Clustering results

Distributed processing

- Centralised processing may be impossible
  - network with communication constraints
  - no node knows fully the signal



- Settings for distributed processing
  - Each node *n* knows
    - its own reading of *y*
    - the *n*th row of the Laplacian
    - the coefficients used in the dictionary
  - Signal is processed distributively

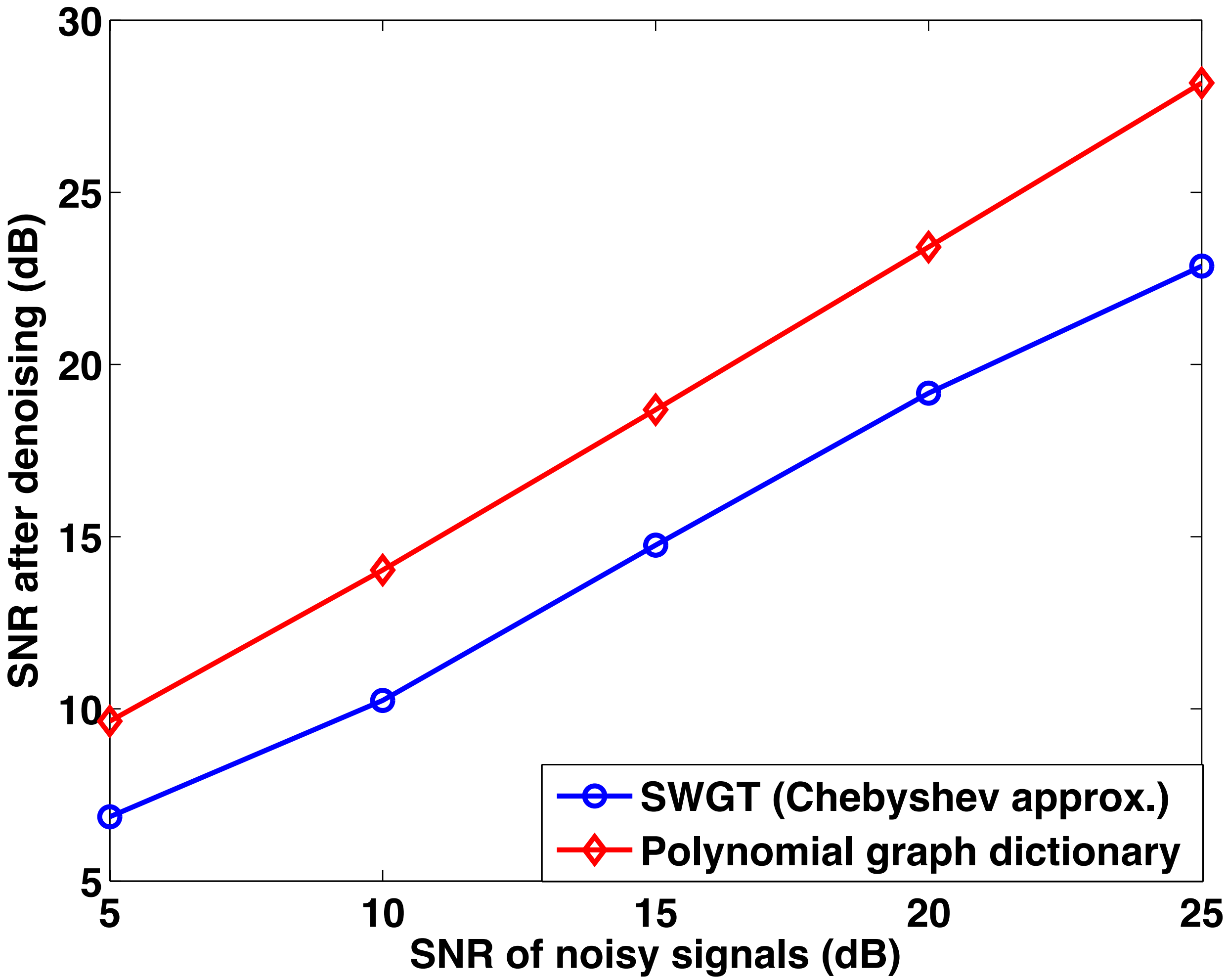
Good news: the spectral dictionary can be distributed!

Distributed processing of adjoint

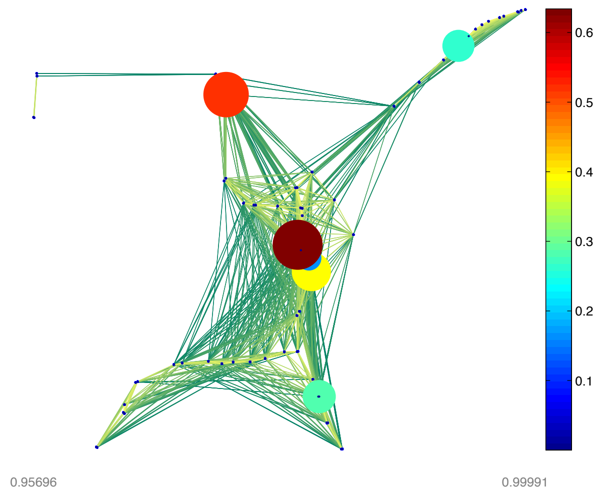
Algorithm 1 Distributed computation of $\mathcal{D}^T y$

Inputs at node n**:** $y(n), \mathcal{L}_{n,:}, \alpha = [\alpha_1; ...; \alpha_S]$
Output at node n**:** $\{(\mathcal{D}^T y)_{(s-1)N+n}\}_{s=1,..,S}$
Transmit $y(n)$ to all neighbors $\mathcal{N}_n$
Receive $y(m)$ from neighbors $\mathcal{N}_n$
Compute and store $c_n^1 = (\mathcal{L}^T y)_n$.
for $k = 2, ..., K$ **do:**
    Transmit $c_n^{k-1} = (\mathcal{L}^T c^{k-2})_n$ to all the neighbors
    Receive c_m^{k-1} from all the neighbors $m \in \mathcal{N}_n$.
end for
for $s = 1, .., S$ **do**
    Compute $(\mathcal{D}^T y)_{(s-1)N+n} = \alpha_{0s} y(n) + \sum_{k=1}^{K} \alpha_{ks} c_n^k$
end for

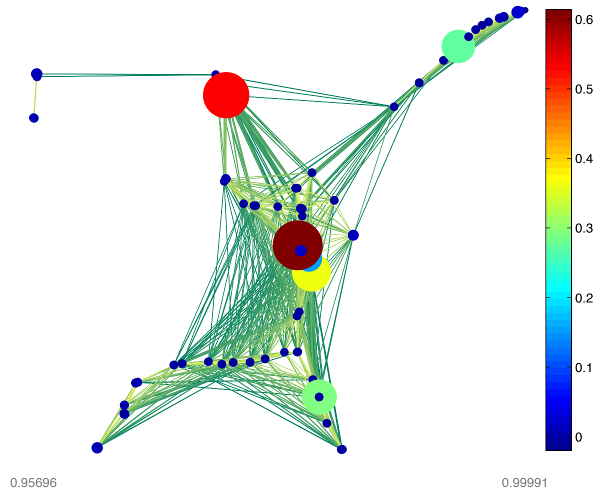
Denoising experiments



Distributed denoising with 100 ISTA iterations



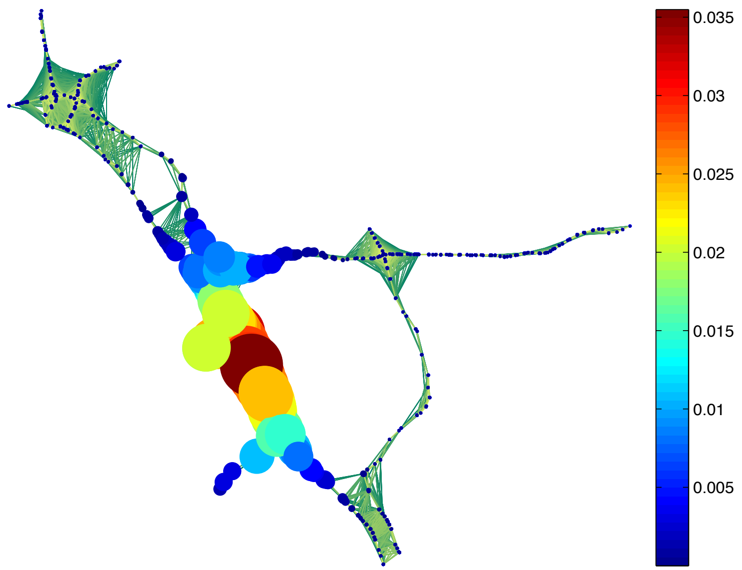
Clean traffic bottleneck signal

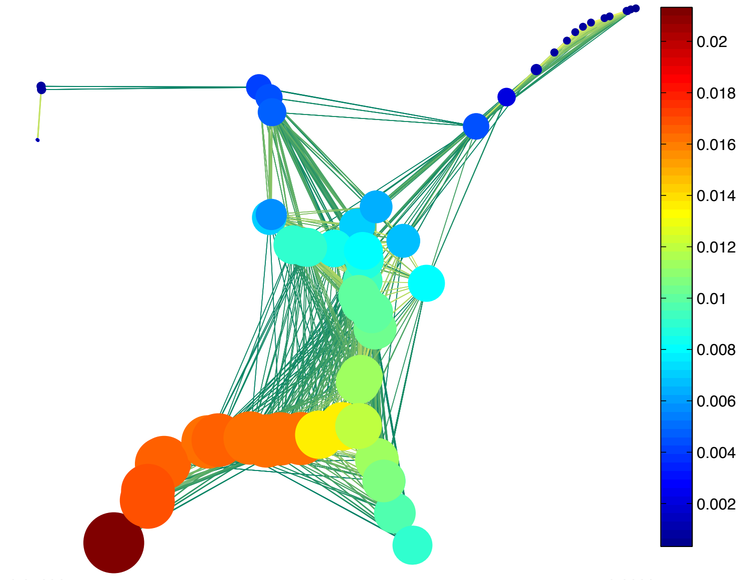


Denoised traffic bottleneck signal [24 dB]

Next? Signals on Multiple Graphs

- A process could be observed on different graphs (e.g., traffic bottlenecks in different cities)





- The evolution of the process depends on the graph: the observations may be visually different

Graph Signal Model

- We consider graph signals that are linear combinations of a few overlapping local processes at different nodes (localized patterns)

- Given a set of processes $\mathcal{P} = \{g_s(\mathcal{G})\}_{s=1}^{S}$, a signal y on a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ can be decomposed as:

$$y = \sum_{g \in \widetilde{\mathcal{P}}, n \in \widetilde{\mathcal{V}}} y_{g,n}, \quad \text{where } \widetilde{\mathcal{P}} \subseteq \mathcal{P}, \widetilde{\mathcal{V}} \subseteq \mathcal{V}$$

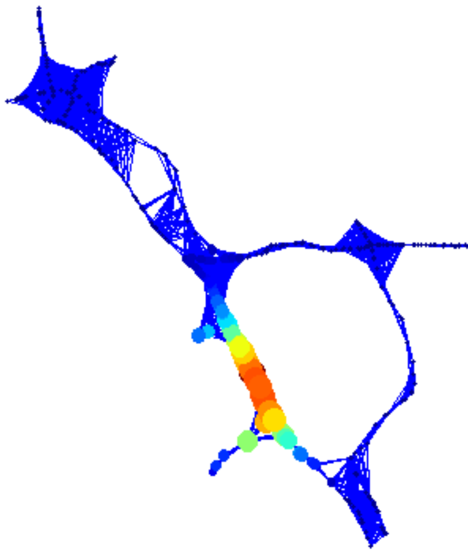
sparse set of components

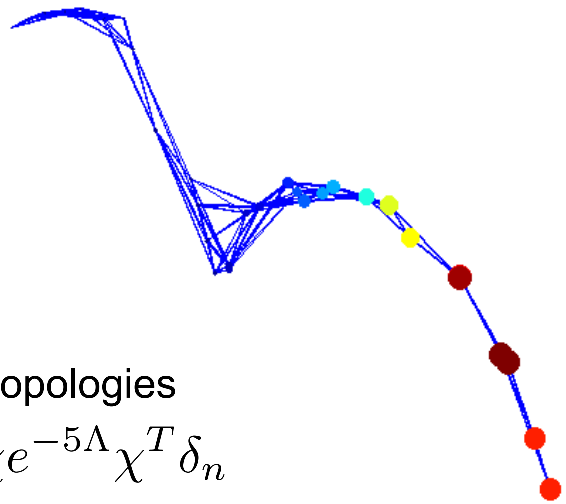
- The same processes may evolve across different topologies

Multi-graph dictionary learning

- **Problem:** Learn atoms for effective representation of signals, that are collected on different graph topologies

- **Main assumption:** Signals on different topologies may share similar spectral characteristics





- same process evolving in two different topologies
- signal observation: $y = e^{-5\mathcal{L}}\delta_n = \chi e^{-5\Lambda}\chi^T\delta_n$

Multi-Graph Dictionary Learning Problem

- Given a set of training signals $Y_t = [y_{t1}, y_{t2}, ..., y_{tM_t}]$, living on the weighted graphs $\mathcal{G}_t, t = \{1, 2, ..., T\}$, solve:

$$\underset{\alpha \in \mathbb{R}^{(K+1)S},\ X_t \in \mathbb{R}^{SN \times M_t}}{\text{argmin}} \left\{ \sum_{t=1}^{T} \frac{1}{M_t} ||Y_t - \mathcal{D}_t X_t||_F^2 + \mu ||\alpha||_2^2 \right\}$$

$$\text{subject to} \quad \|X_t^m\|_0 \leq \gamma, \quad \forall m \in \{1, ..., M_t\},$$

$$\mathcal{D}_t^s = \sum_{k=0}^{K} \alpha_{sk} \mathcal{L}_t^k, \forall s \in \{1, 2, ..., S\},$$

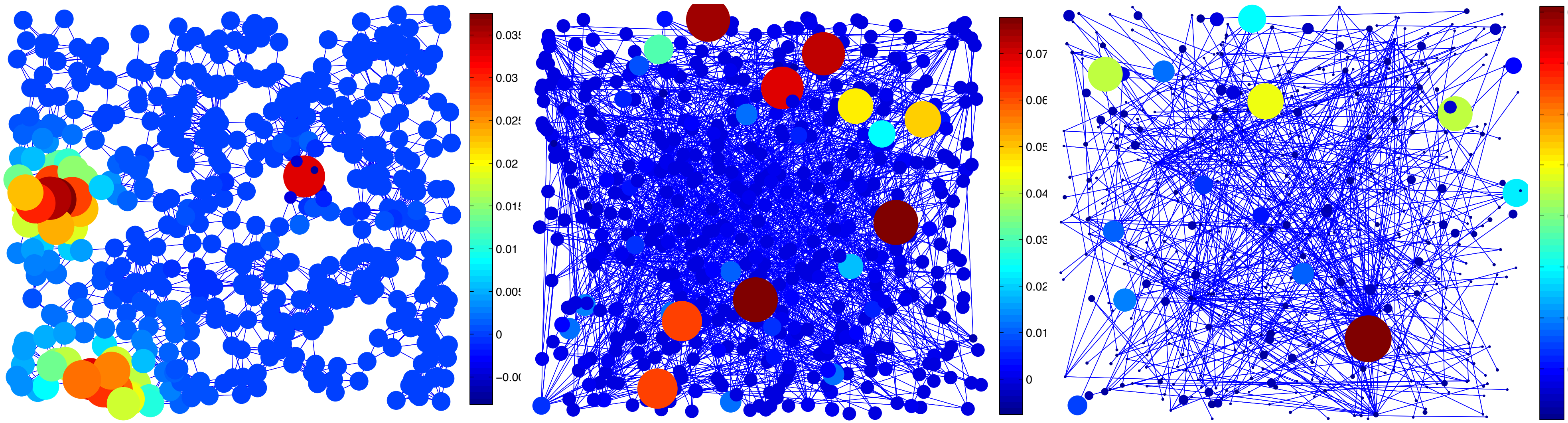
$$0 \preceq \mathcal{D}_t^s \preceq c, \forall s \in \{1, 2, ..., S\},$$

Each subdictionary captures the same process evolving in different topologies

Same polynomial coefficients for all topologies

Synthetic Experiments

- Generate 3 graphs $\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3$ of 500 nodes



(a) RBF model $\mathcal{G}_1$

(b) Forest Fire model $\mathcal{G}_2$

(c) Barabasi-Albert model $\mathcal{G}_3$

Synthetic Graph Processes

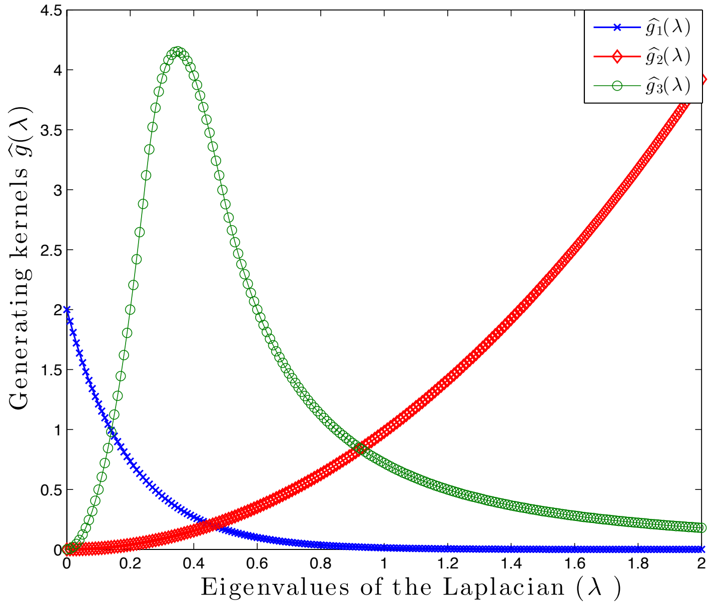
- Consider 3 subdictionaries generated from the following processes:
  1) Heat diffusion kernel $\widehat{g}_1(\lambda) = e^{-5\lambda}$ → $\mathcal{D}^1 = \chi \widehat{g}_1(\Lambda) \chi^T$
  2) Wave kernel $\widehat{g}_2(\lambda) = e^{-(0.01-\log\lambda)^2}$ → $\mathcal{D}^2 = \chi \widehat{g}_2(\Lambda) \chi^T$
  3) Spectral graph wavelet kernel (bandpass filter)
     $\widehat{g}_3(\lambda) = \widehat{g}(4.1\lambda)$ → $\mathcal{D}^3 = \chi \widehat{g}_3(\Lambda) \chi^T$

- Training signals: linear combination of a few atoms of the above subdictionaries on one graph

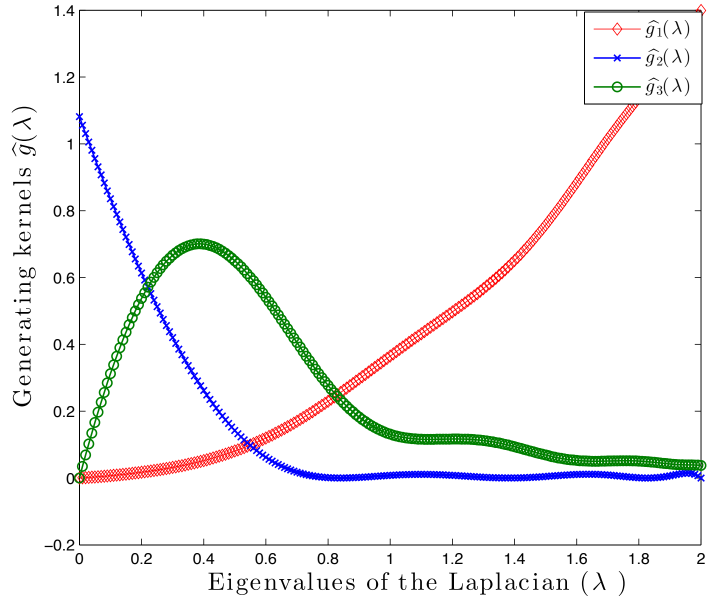
$$y_t = [\mathcal{D}_t^1 \ \mathcal{D}_t^2 \ \mathcal{D}_t^3]x, \quad \text{where} \quad \|x\|_0 \leq 4$$

Recovery of Graph Processes

- Processes are learned jointly from 1200 training signals on the three graphs $(M = M_1 = M_2 = M_3 = 400)$



Original kernels

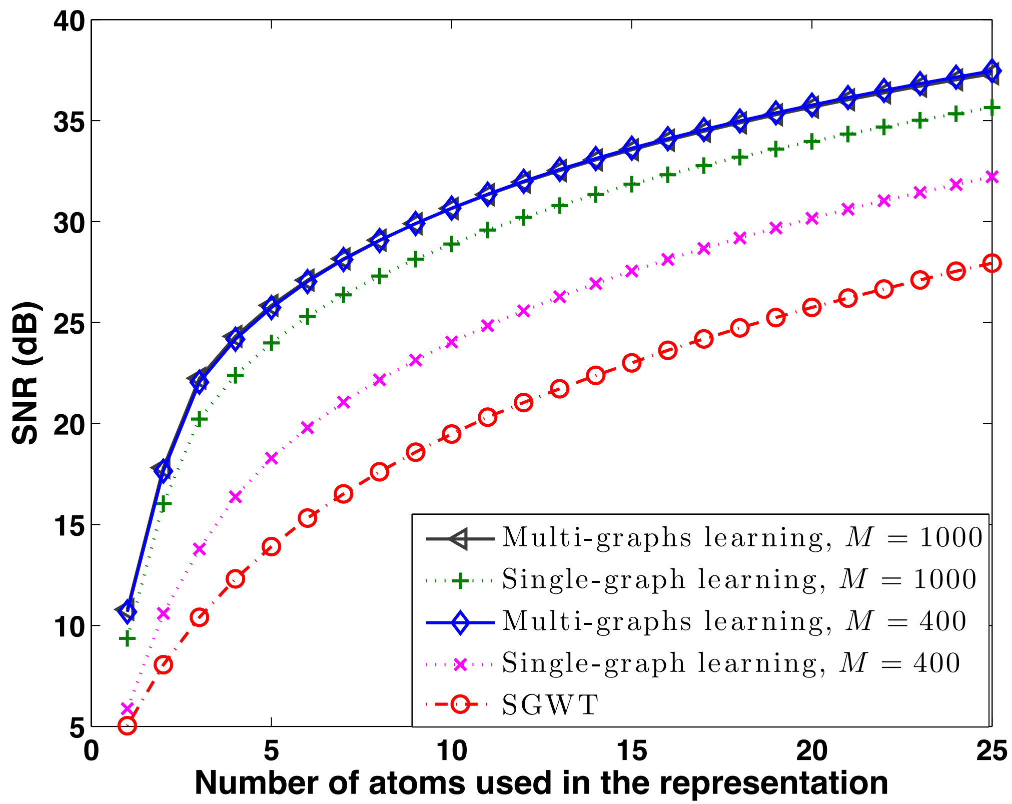


Learned kernels

- The proposed algorithm is able to recover the continuous processes

Representation of Synthetic Signals

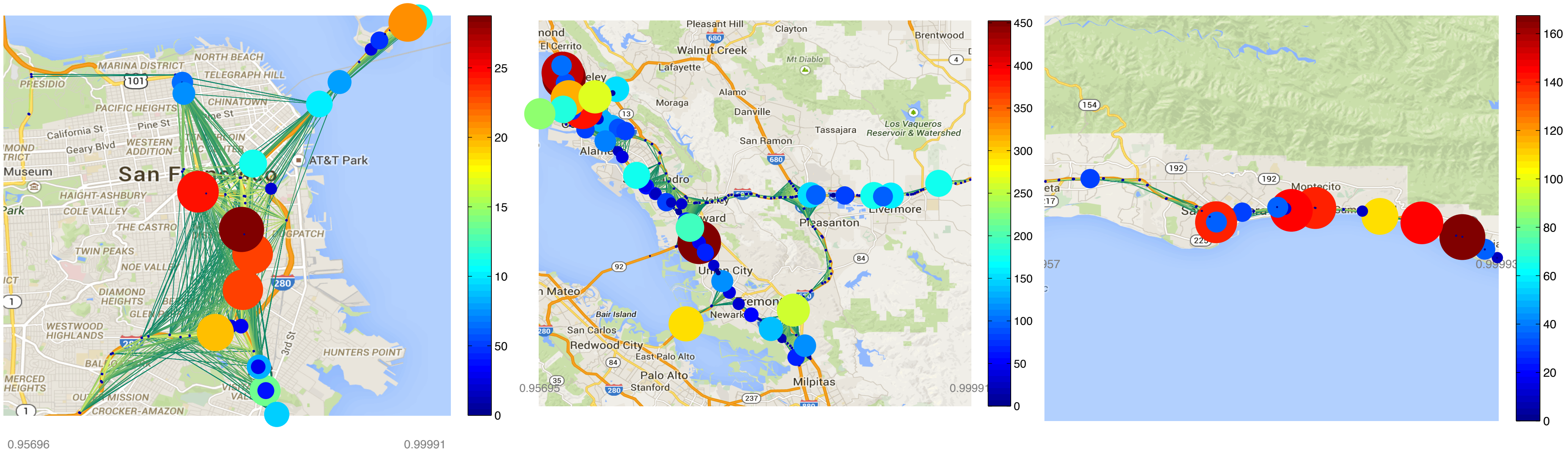
- Learn a dictionary for different sizes of the training set



- Joint learning compensates for the lack of training signals in each graph separately

Representation of Traffic Signals

- Consider bottleneck signals[1] from Jan. 2007-Aug.2014 on three different graphs:

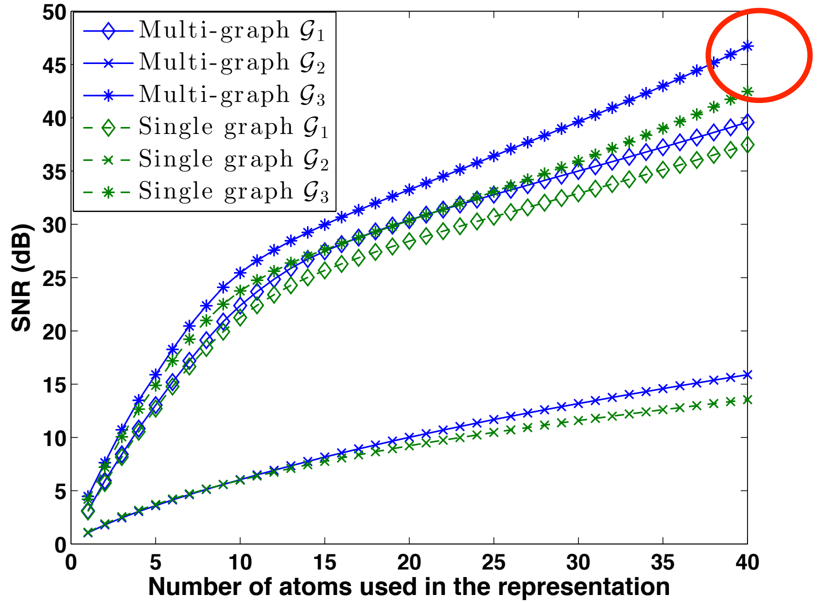


(a) San Francisco ($\mathcal{G}_1$) **(b) Alameda** ($\mathcal{G}_2$) **(c) Santa Barbara** ($\mathcal{G}_3$)

[1]The data are publicly available at http://pems.dot.ca.gov.

Representation of Traffic Signals

- Learn a dictionary from training signals on different graphs: 1383 signals in San Francisco, 1386 in Alameda, 447 in Santa Barbara
- Approximate with the learned kernels testing signals on specific graph



- Joint learning outperforms independent learning on each graph

Summary

- Take-home messages:
  - Graph signal processing is a very generic and promising framework
  - Polynomial matrix functions of the graph Laplacian seems to be a flexible structure for sparsely representing graph signals
  - Polynomial kernels lead to effective implementations

- Still many open questions:
  - Development of applications where the kernel information could be beneficial, such as classification, learning, etc
  - Limits of multi-graph dictionary learning
  - Definition of the optimal graph topology

Acknowledgments

- Dorina Thanou, EPFL
- Xiaowen Dong, MIT
- David Shuman, Macalester College
- Pierre Vandergheynst, EPFL
- Antonio Ortega, USC
- Sunil Narang, Bing
- Phil Chou, MSR
- many others…

References

- D. I Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, “The emerging field of signal processing on graphs: Extending high - dimensional data analysis to networks and other irregular domains,” IEEE Signal Process. Mag., vol. 30, no. 3, pp. 83–98, May 2013
- **D. Thanou, D. I Shuman, and P. Frossard, “Learning Parametric Dictionaries for Signals on Graphs”, IEEE Trans. Signal Process., vol. 62, no. 15, Aug. 2014**
- **D. Thanou, and P. Frossard, “Multi-graph learning of spectral graph dictionaries”, accepted in ICASSP 2015.**
- D. Hammond, P. Vandergheynst, and R. Gribonval, “Wavelets on graphs via spectral graph theory,” Appl. Comput. Harmon. Anal., vol. 30, no. 2, pp. 129–150, March 2010.
- X. Zhang, X. Dong, and P. Frossard, “Learning of structured graph dictionaries,” in Proc. IEEE Int. Conf. Acc., Speech, and Signal Process., Kyoto, Japan, Mar. 2012, pp. 3373 – 3376.