

# AIMS CDT - Signal Processing

## Michaelmas Term 2022

Xiaowen Dong

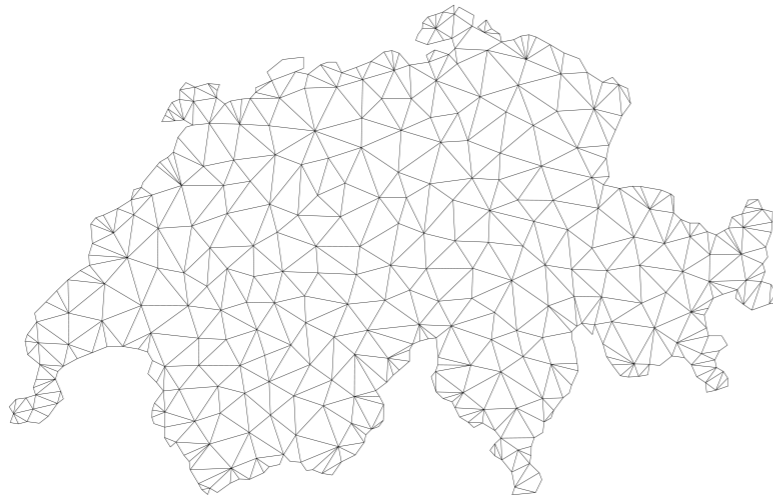
Department of Engineering Science



# Introduction to Graphs

## Signal Processing

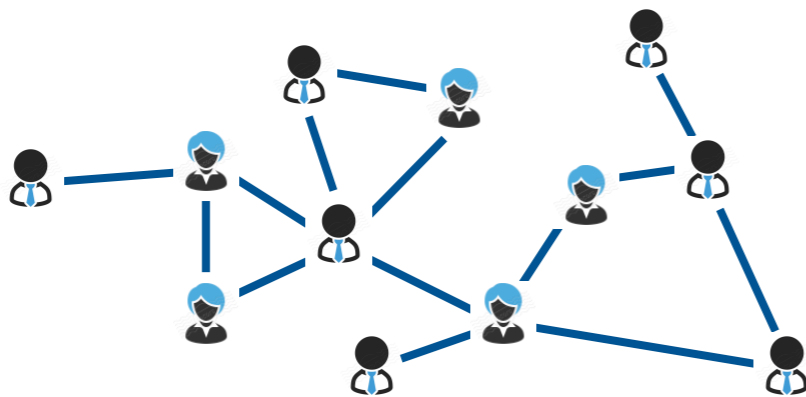
# Graphs are everywhere



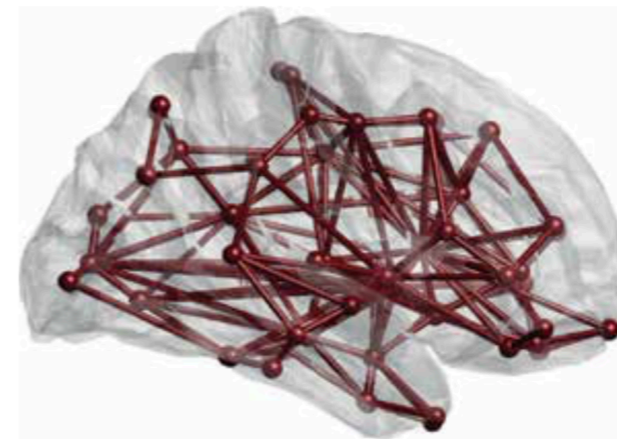
**geographical network**



**traffic network**

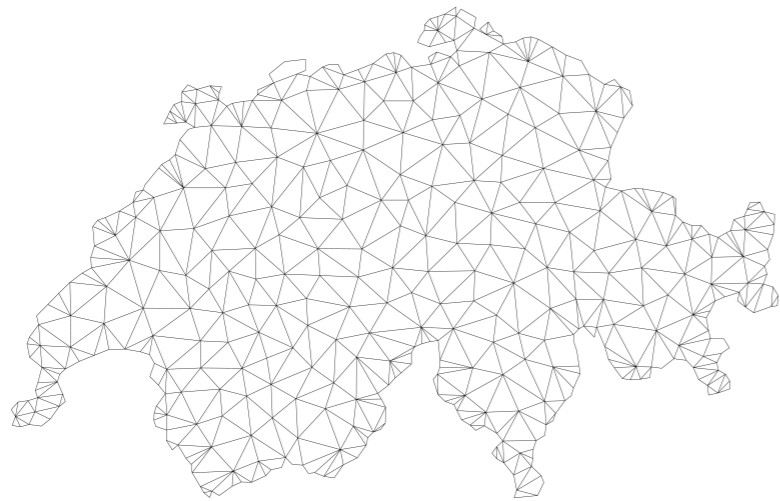


**social network**



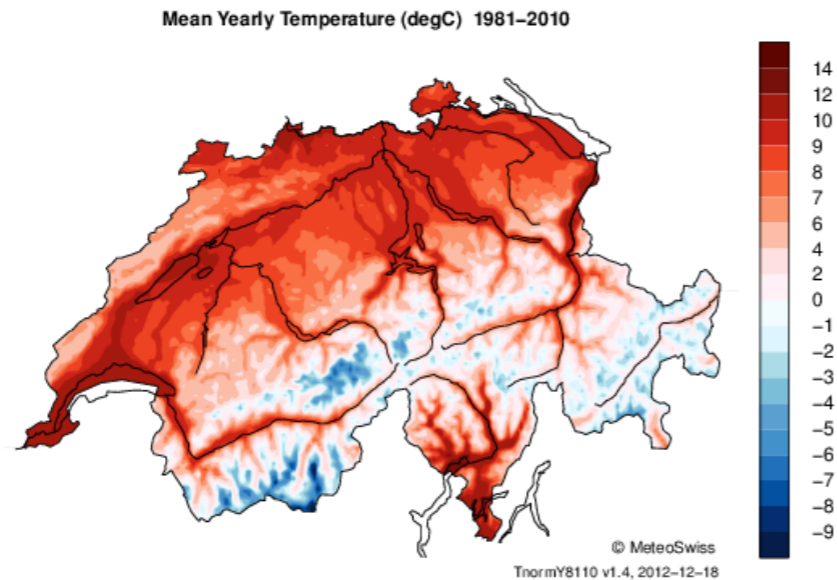
**brain network**

# Graph-structured data are everywhere



- vertices
  - geographical regions
- edges
  - geographical proximity between regions

# Graph-structured data are everywhere



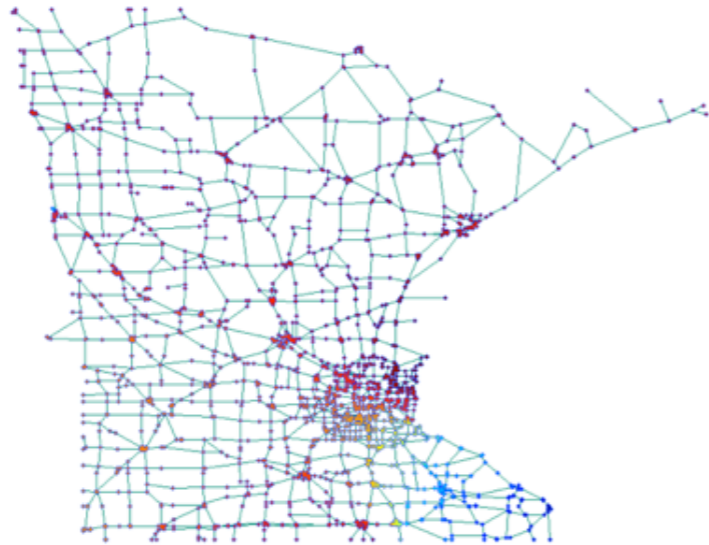
- vertices
  - geographical regions
- edges
  - geographical proximity between regions
- signal
  - temperature records in these regions

# Graph-structured data are everywhere



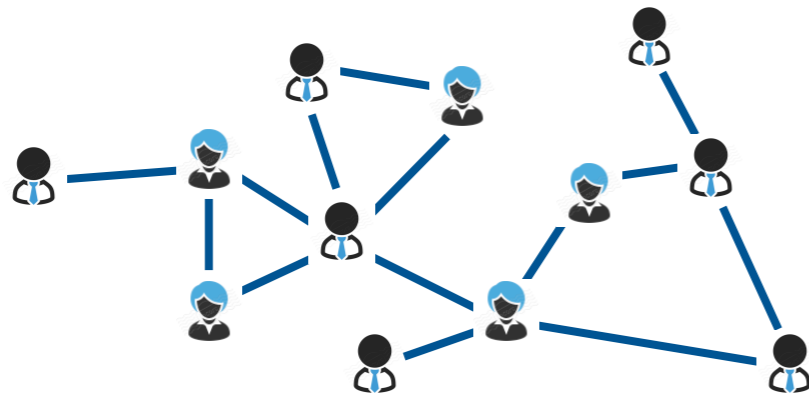
- vertices
  - road junctions
- edges
  - road connections

# Graph-structured data are everywhere



- vertices
  - road junctions
- edges
  - road connections
- signal
  - traffic congestion at junctions

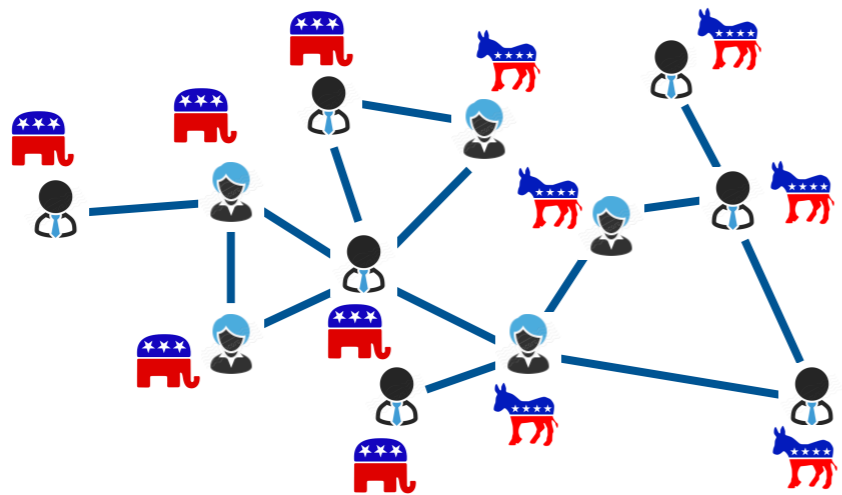
# Graph-structured data are everywhere



- vertices
  - individuals
- edges
  - friendship between individuals

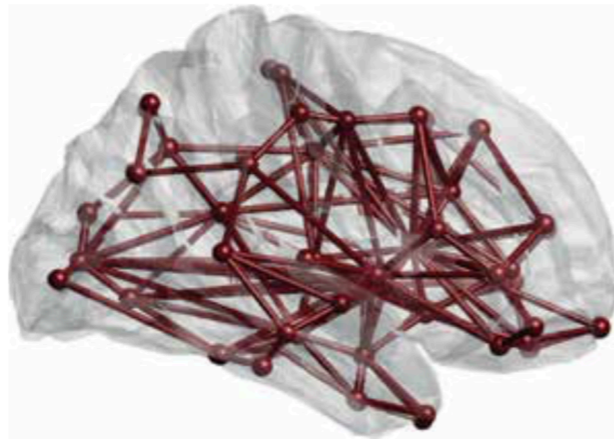


# Graph-structured data are everywhere



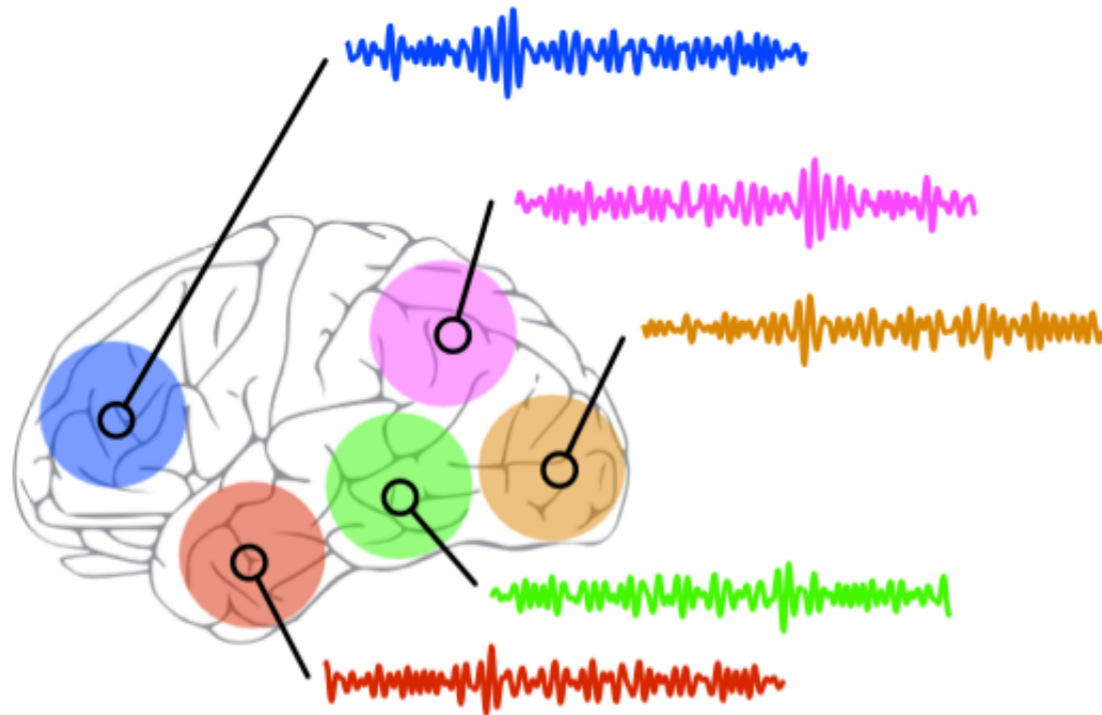
- vertices
  - individuals
- edges
  - friendship between individuals
- signal
  - political view

# Graph-structured data are everywhere



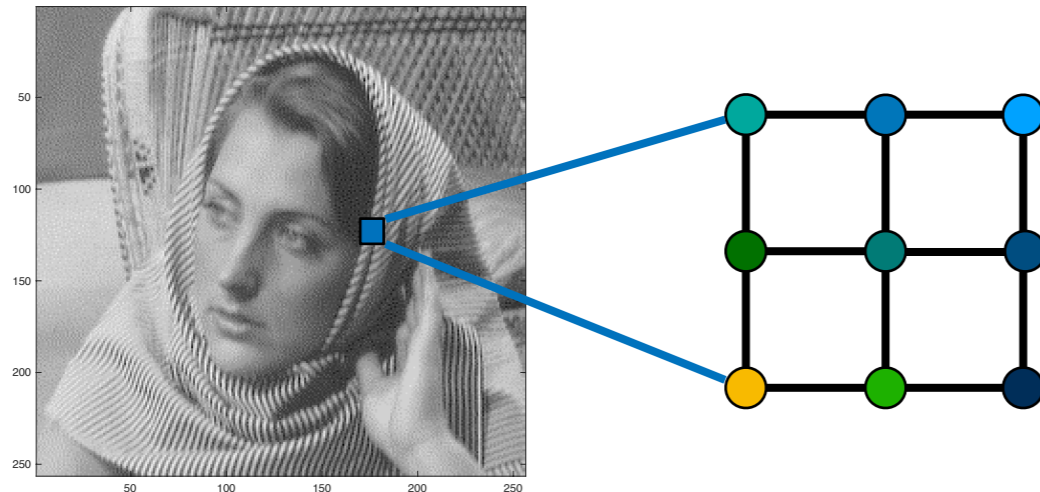
- vertices
  - brain regions
- edges
  - structural connectivity between brain regions

# Graph-structured data are everywhere



- vertices
  - brain regions
- edges
  - structural connectivity between brain regions
- signal
  - blood-oxygen-level-dependent (BOLD) time series

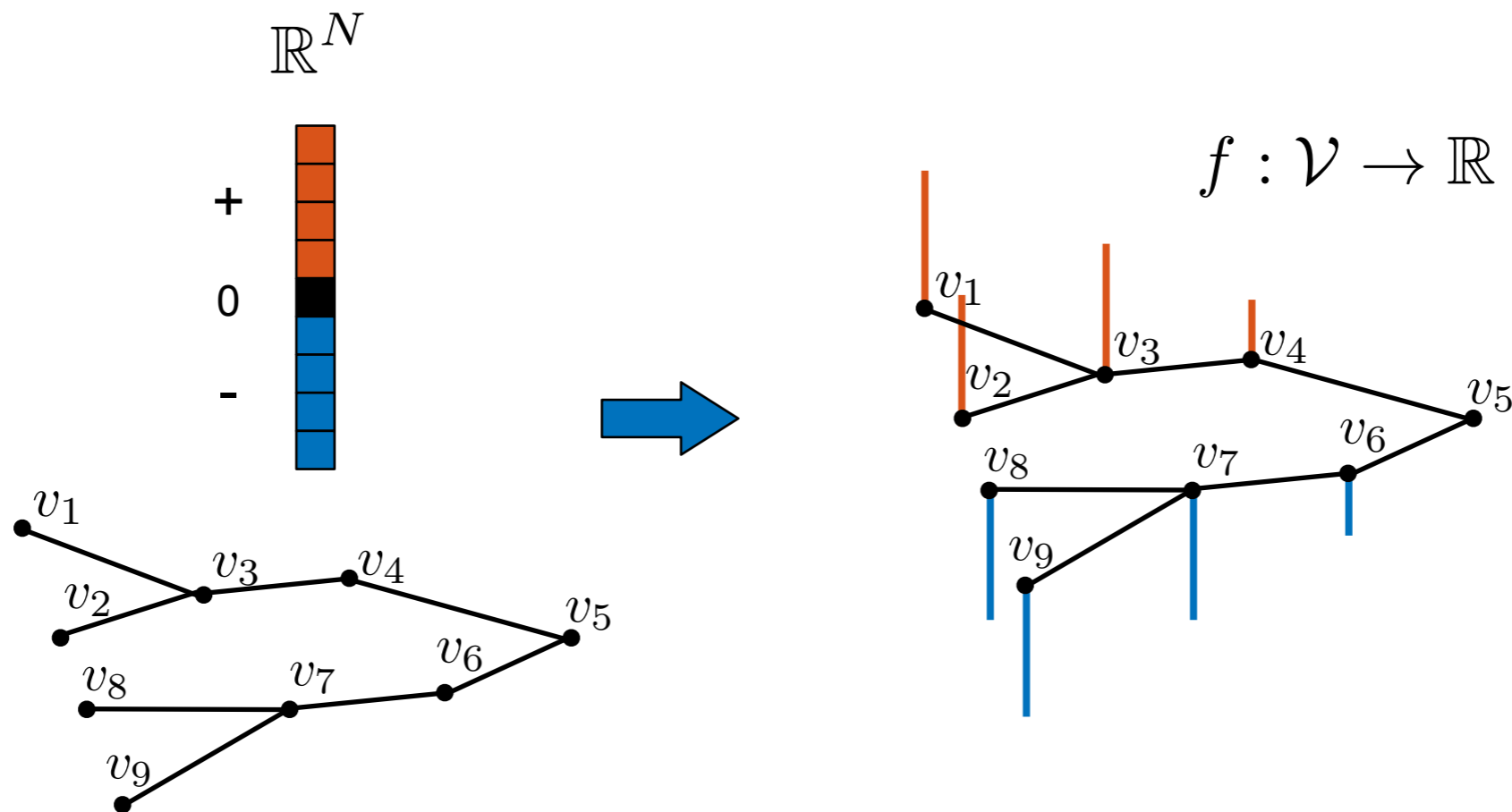
# Graph-structured data are everywhere



- nodes
  - pixels
- edges
  - spatial proximity between pixels
- signal
  - pixel values

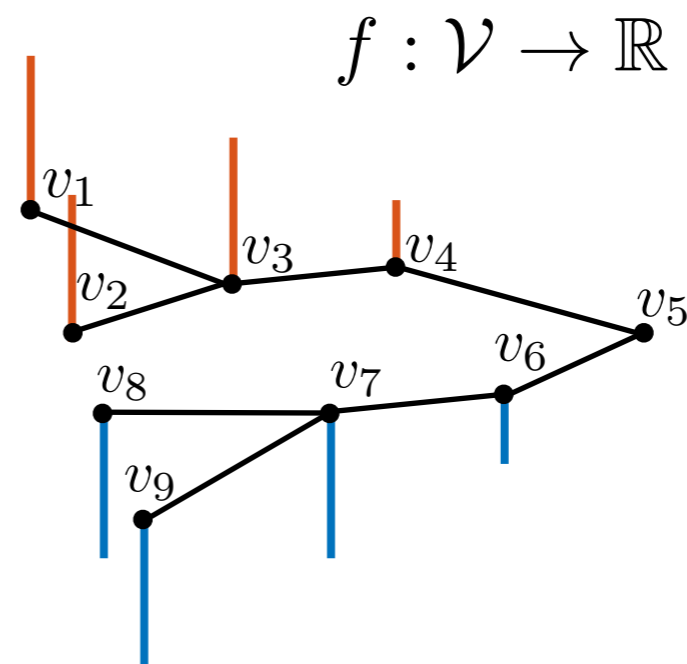
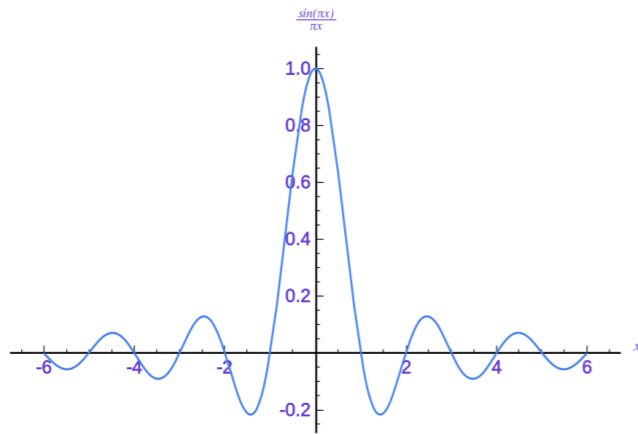
# Graph signal processing

- Graph-structured data can be represented by signals defined on graphs or **graph signals**



takes into account both structure (edges) and data (values at vertices)

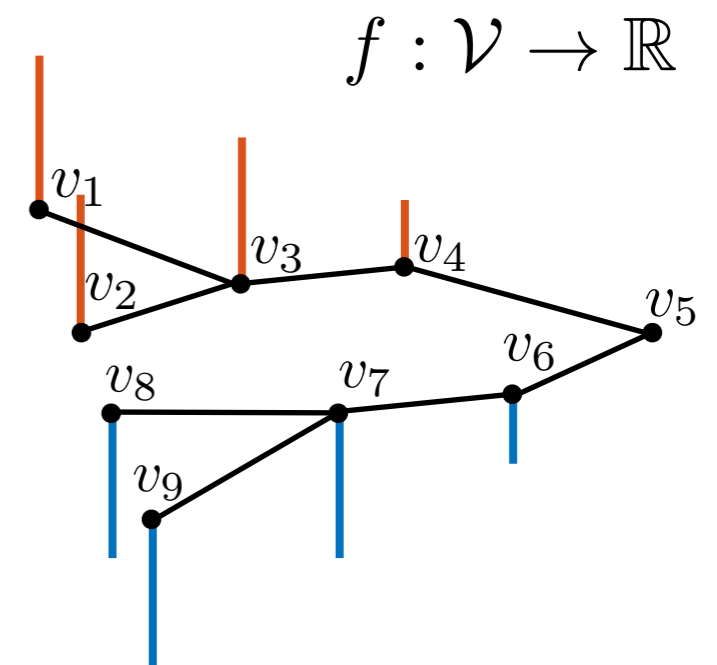
# Graph signal processing



how to generalise classical signal processing tools  
on irregular domains such as graphs?

# Graph signal processing

- Graph signals provide a nice compact format to encode structure within data
- Generalisation of classical signal processing tools can greatly benefit analysis of such data
- Numerous applications: Transportation, biomedical, social, economic network analysis
- An increasingly rich literature
  - classical signal processing
  - algebraic and spectral graph theory
  - computational harmonic analysis
  - machine learning



# Outline

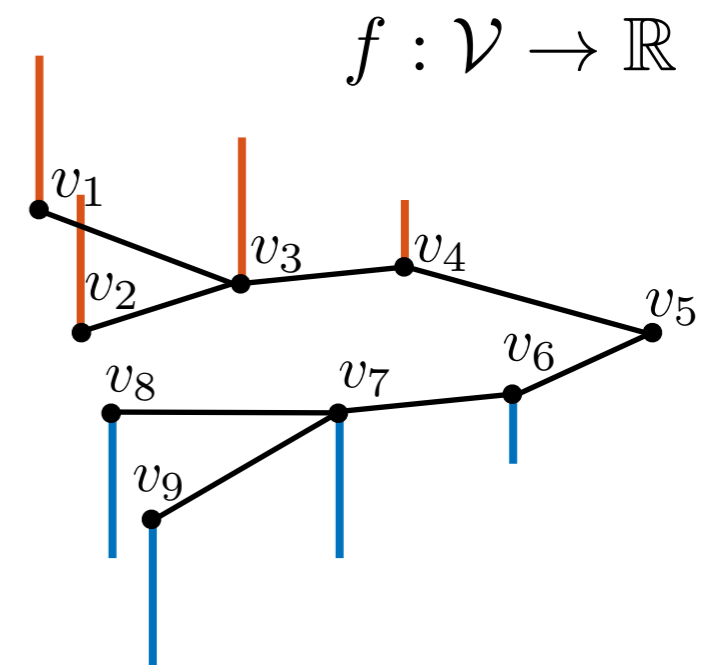
- Graph signal processing (GSP): Basic concepts
- Graph spectral filtering: Basic tools of GSP
- Connection with literature
- Representation of graph signals
- Applications



# Two paradigms

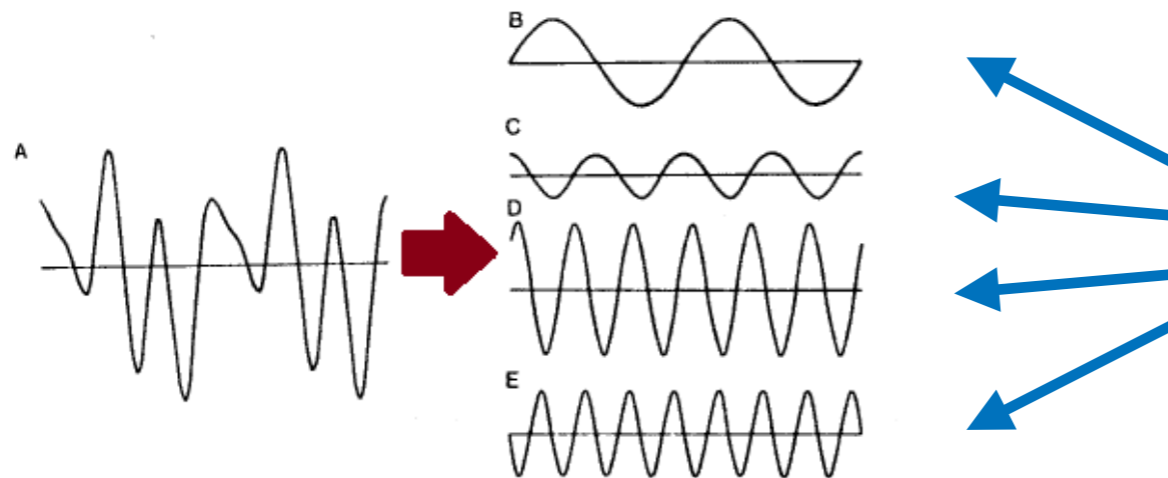
- Main GSP approaches can be categorised into two families:
  - vertex (spatial) domain designs
  - frequency (graph spectral) domain designs

**important for analysis of signal properties**



# Need for notion of frequency

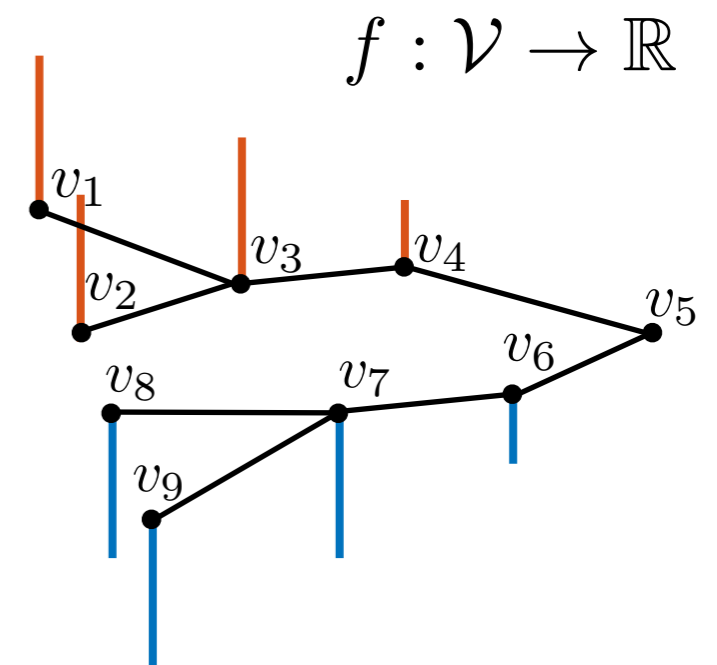
- Classical Fourier transform provides frequency domain representation of signals



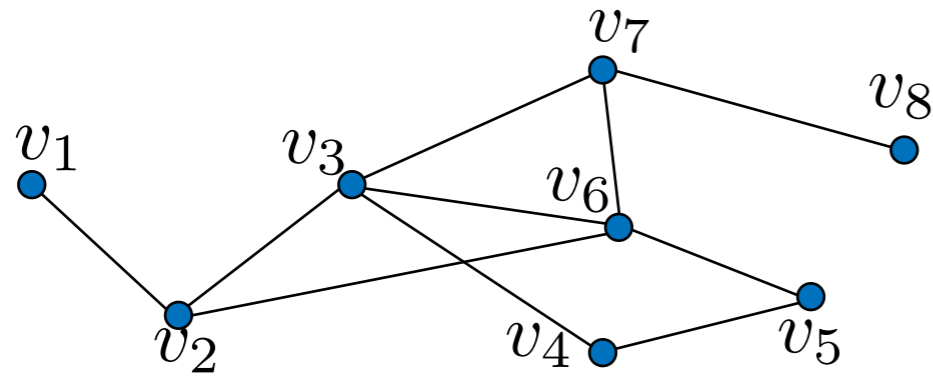
- “building blocks” of signal
- different frequency (oscillation)

- What about a notion of frequency for graph signals?

**we need the graph Laplacian matrix**



# Graph Laplacian



weighted and undirected graph:

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

$$D = \text{diag}(d(v_1), \dots, d(v_N))$$

$$L = D - W \quad \text{equivalent to } W!$$

$$L_{\text{norm}} = D^{-\frac{1}{2}} (D - W) D^{-\frac{1}{2}}$$

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

$D$

$W$

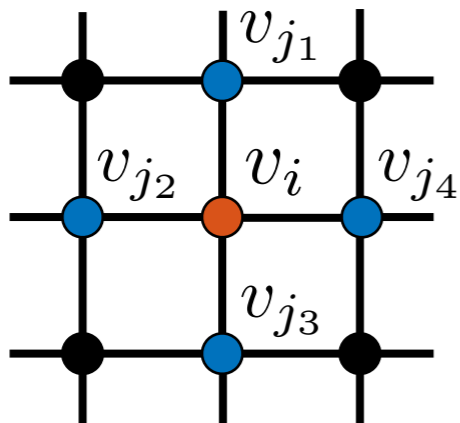
$L$

- symmetric
- off-diagonal entries non-positive
- rows sum up to zero

# Graph Laplacian

## Why graph Laplacian?

- approximation of the Laplace operator

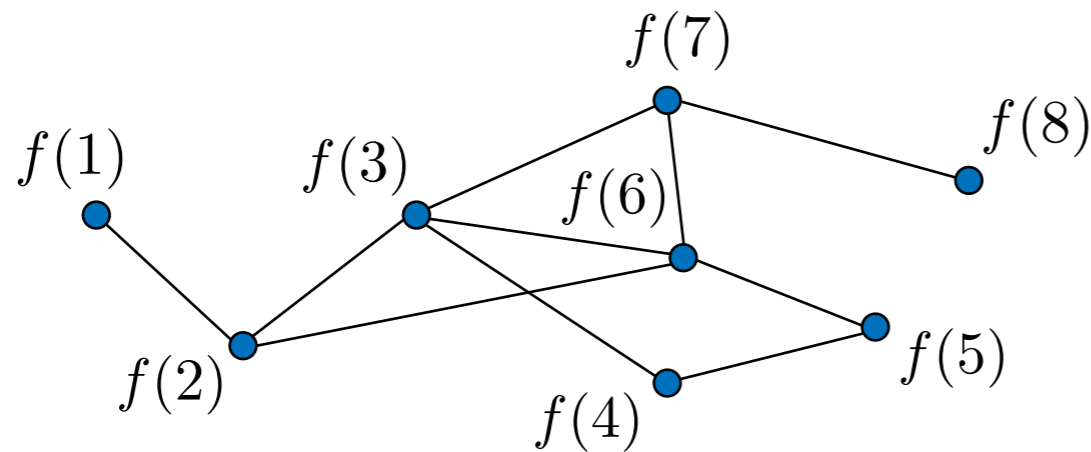


$$(Lf)(i) = (4f(i) - f(j_1) - f(j_2) - f(j_3) - f(j_4))/(\delta x)^2$$

standard 5-point stencil for approximating  $-\nabla^2 f$

- converges to the Laplace-Beltrami operator (given certain conditions)
- provides a notion of “frequency” on graphs

# Graph Laplacian



graph signal  $f : \mathcal{V} \rightarrow \mathbb{R}$

$$\begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & -1 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & -1 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} f(1) \\ f(2) \\ f(3) \\ f(4) \\ f(5) \\ f(6) \\ f(7) \\ f(8) \end{pmatrix}$$

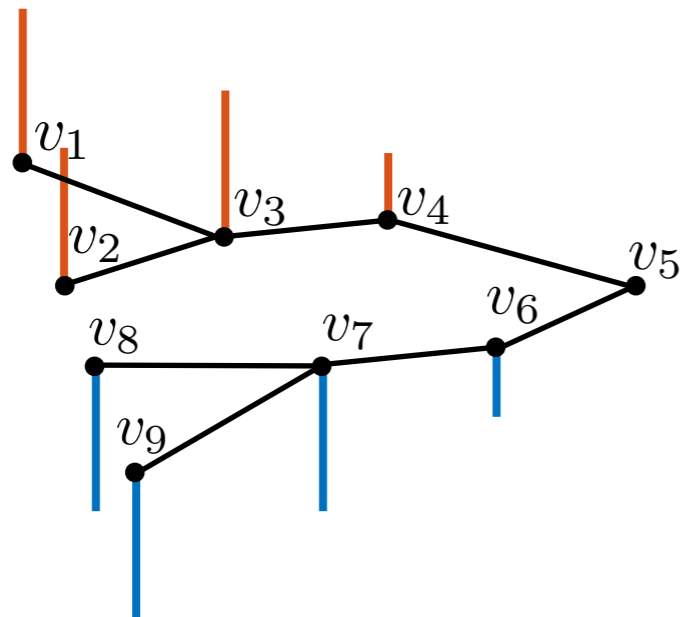
$$\begin{pmatrix} f(1) \\ f(2) \\ f(3) \\ f(4) \\ f(5) \\ f(6) \\ f(7) \\ f(8) \end{pmatrix}^T \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & -1 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & -1 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} f(1) \\ f(2) \\ f(3) \\ f(4) \\ f(5) \\ f(6) \\ f(7) \\ f(8) \end{pmatrix}$$

$$Lf(i) = \sum_{j=1}^N W_{ij} (f(i) - f(j))$$

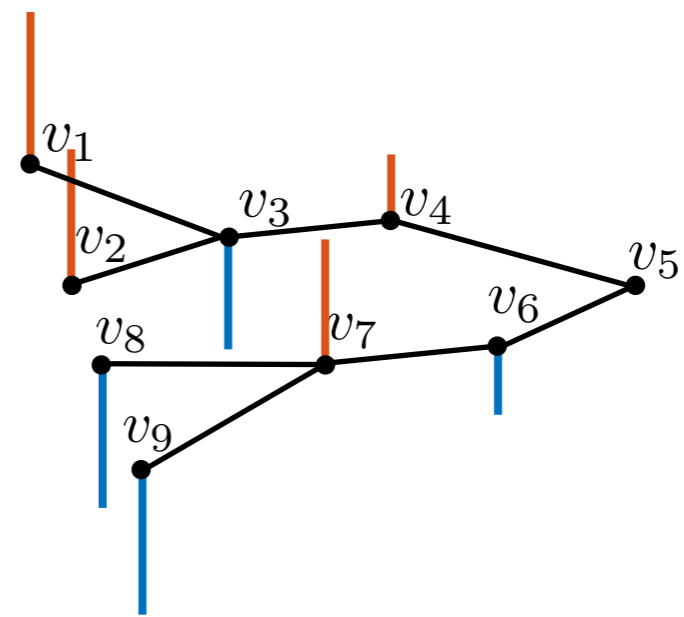
$$f^T Lf = \frac{1}{2} \sum_{i,j=1}^N W_{ij} (f(i) - f(j))^2$$

a measure of “smoothness”

# Graph Laplacian



$$f^T L f = 1$$



$$f^T L f = 21$$

# Graph Laplacian

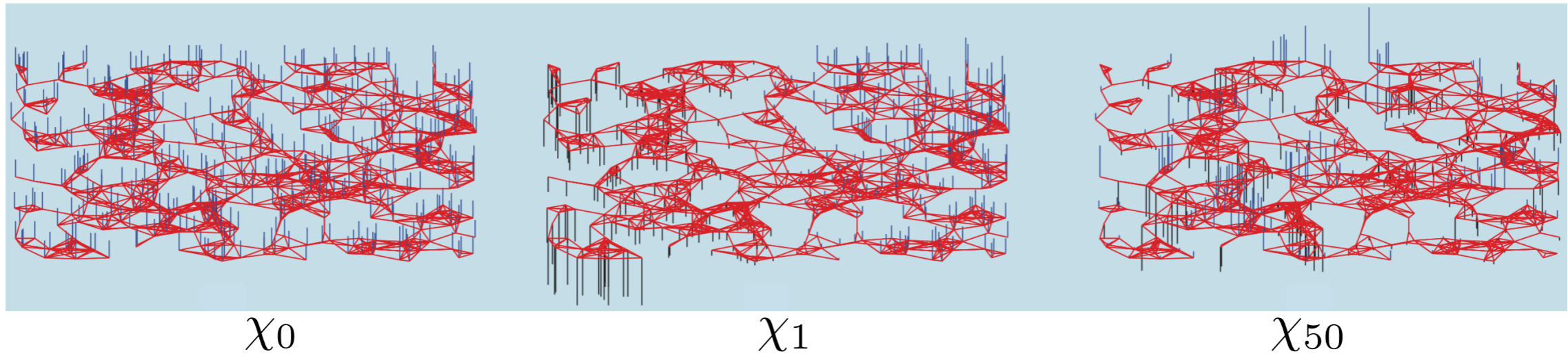
- $L$  has a complete set of orthonormal eigenvectors:  $L = \chi \Lambda \chi^T$

$$L = \begin{bmatrix} | & & | \\ \chi_0 & \cdots & \chi_{N-1} \\ | & & | \end{bmatrix} \begin{bmatrix} \lambda_0 & & 0 \\ & \ddots & \\ 0 & & \lambda_{N-1} \end{bmatrix} \begin{bmatrix} \text{---} \chi_0^T \text{---} \\ \cdots \\ \text{---} \chi_{N-1}^T \text{---} \end{bmatrix}$$

$\chi \qquad \qquad \Lambda \qquad \qquad \chi^T$

- Eigenvalues are usually sorted increasingly:  $0 = \lambda_0 < \lambda_1 \leq \dots \leq \lambda_{N-1}$

# Graph Fourier transform



low frequency

high frequency

$$\chi_0^T L \chi_0 = \lambda_0 = 0$$

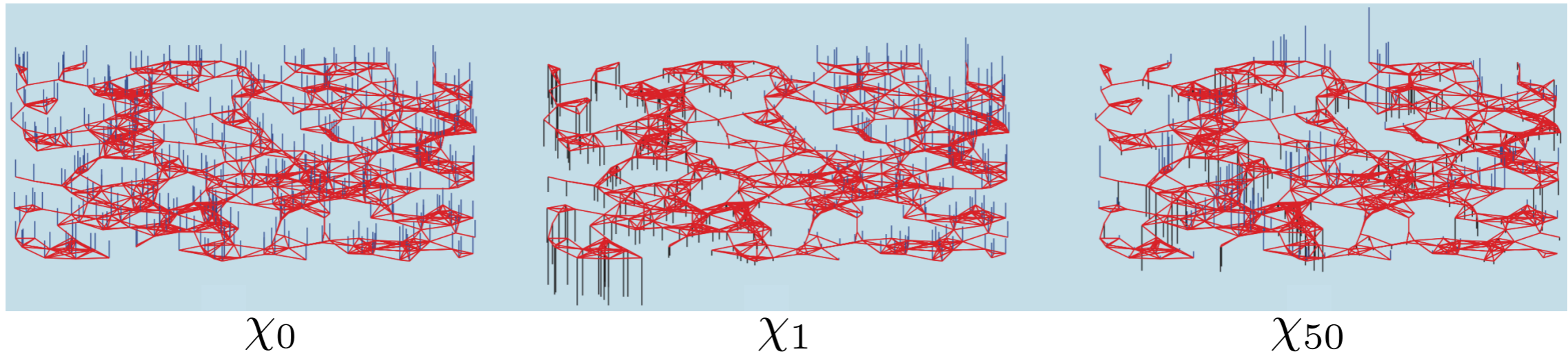
$$\chi_{50}^T L \chi_{50} = \lambda_{50}$$

$$L = \chi \Lambda \chi^T$$

- Eigenvectors associated with smaller eigenvalues have values that vary less rapidly along the edges



# Graph Fourier transform



low frequency

high frequency

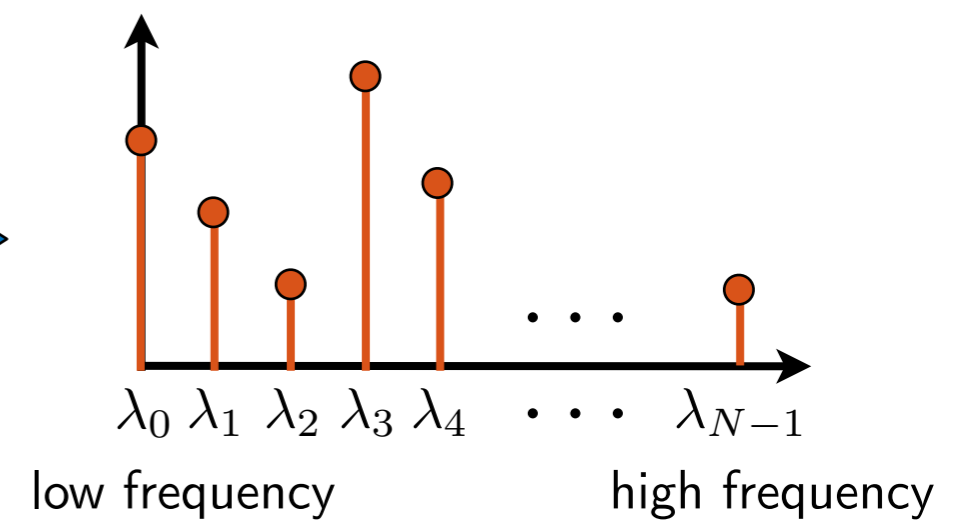
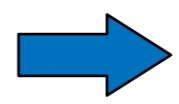
$$\chi_0^T L \chi_0 = \lambda_0 = 0$$

$$\chi_{50}^T L \chi_{50} = \lambda_{50}$$

$$L = \chi \Lambda \chi^T$$

graph Fourier transform:

$$\hat{f}(\ell) = \langle \chi_\ell, f \rangle : \begin{bmatrix} | & & | \\ \chi_0 & \cdots & \chi_{N-1} \\ | & & | \end{bmatrix}^T \begin{bmatrix} | \\ f \\ | \end{bmatrix}$$



# Graph Fourier transform

- The Laplacian  $L$  admits the following eigendecomposition:  $L\chi_\ell = \lambda_\ell\chi_\ell$

one-dimensional Laplace operator:  $-\nabla^2$



eigenfunctions:  $e^{j\omega x}$



Classical FT:  $\hat{f}(\omega) = \int (e^{j\omega x})^* f(x) dx$

$$f(x) = \frac{1}{2\pi} \int \hat{f}(\omega) e^{j\omega x} d\omega$$

graph Laplacian:  $L$



eigenvectors:  $\chi_\ell$

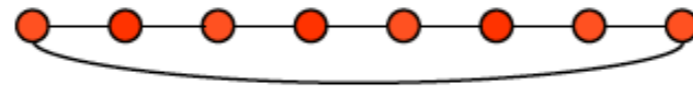


$f : V \rightarrow \mathbb{R}^N$

Graph FT:  $\hat{f}(\ell) = \langle \chi_\ell, f \rangle = \sum_{i=1}^N \chi_\ell^*(i) f(i)$

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

# Two special cases



- (Unordered) Laplacian eigenvalues:  $\lambda_\ell = 2 - 2 \cos\left(\frac{2\ell\pi}{N}\right)$

- One possible choice of orthogonal Laplacian eigenvectors:

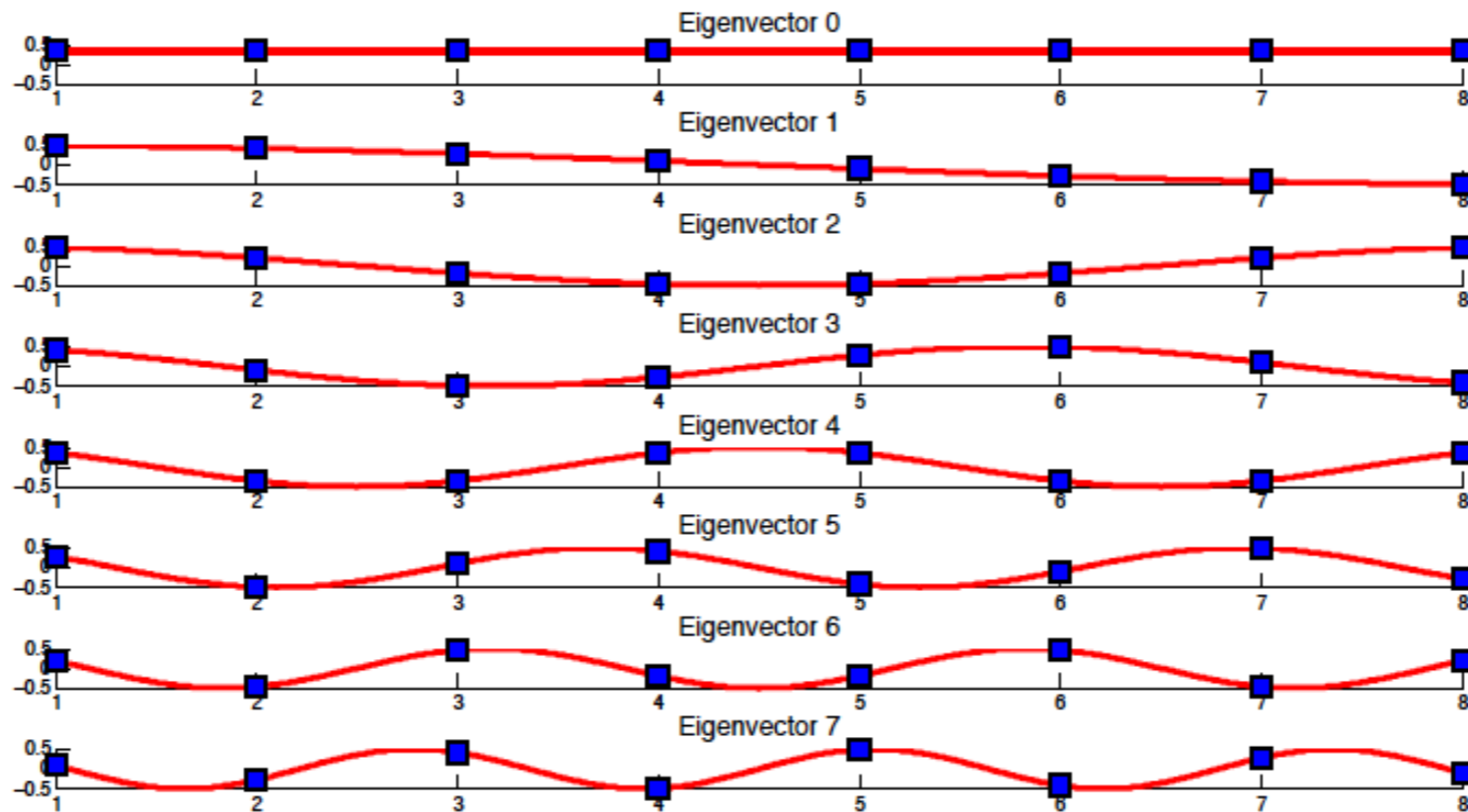
$$\chi_\ell = \left[ 1, \omega^\ell, \omega^{2\ell}, \dots, \omega^{(N-1)\ell} \right], \text{ where } \omega = e^{\frac{2\pi j}{N}}$$

- $\begin{bmatrix} | & & | \\ \chi_0 & \cdots & \chi_{N-1} \\ | & & | \end{bmatrix}$  is the Discrete Fourier Transform (DFT) matrix

# Two special cases



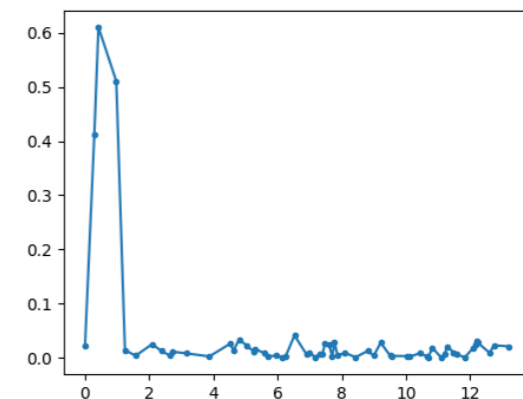
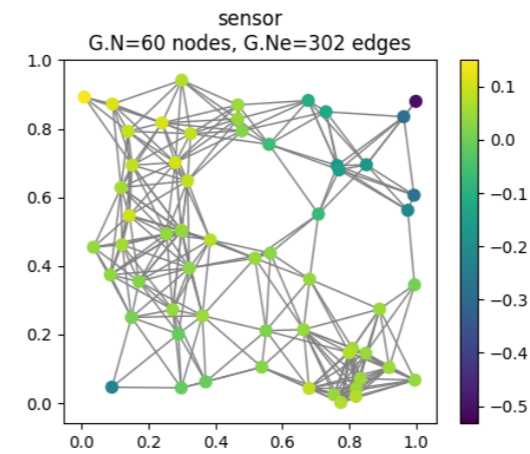
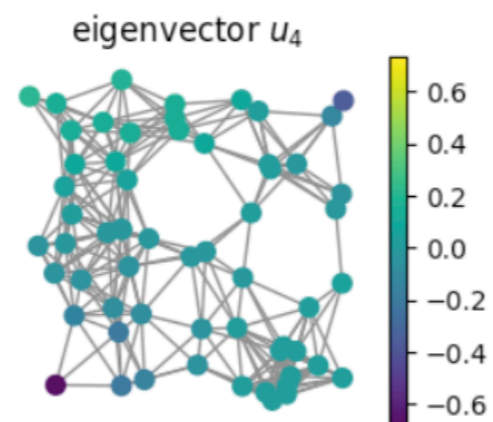
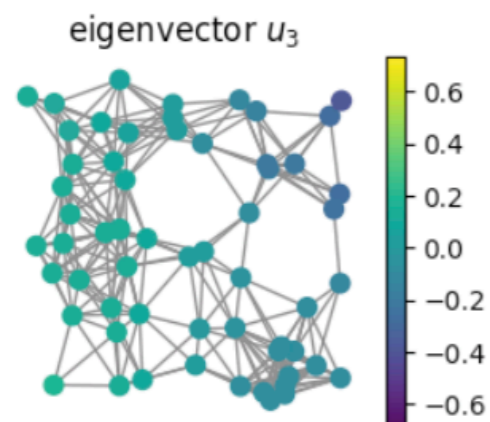
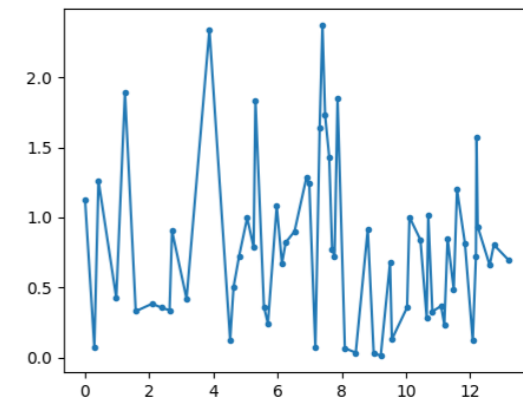
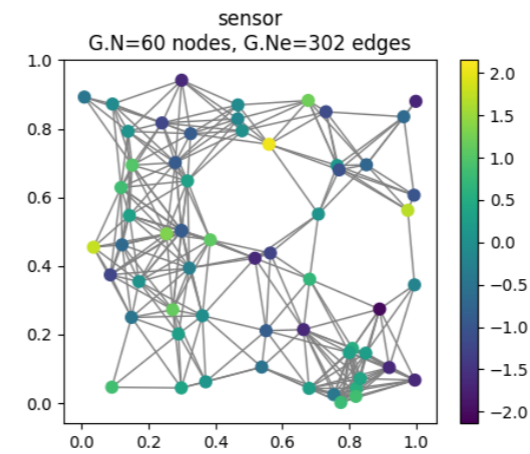
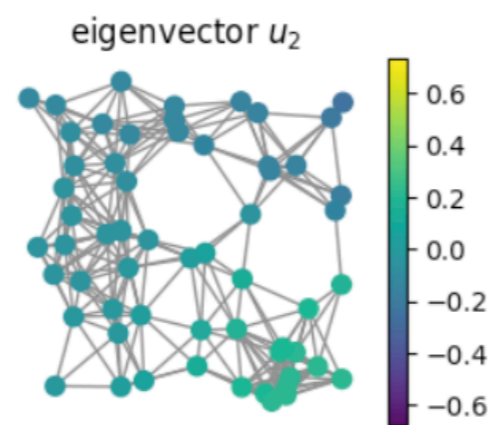
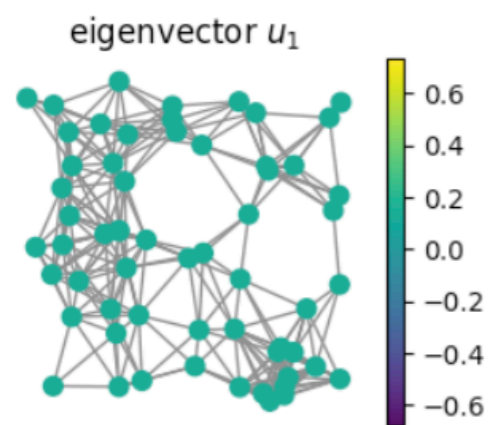
$$\lambda_\ell = 2 - 2 \cos\left(\frac{\pi\ell}{N}\right) \quad \chi_0(i) = \frac{1}{\sqrt{N}}, \quad \chi_\ell(i) = \sqrt{\frac{2}{N}} \cos\left(\frac{\pi\ell(i-0.5)}{N}\right), \quad \ell = 1, 2, \dots, N-1$$



$$\begin{bmatrix} | & & | \\ \chi_0 & \cdots & \chi_{N-1} \\ | & & | \end{bmatrix}$$
 is the Discrete Cosine Transform matrix (DCT-II, Strang, 1999), which is used in JPEG image compression

# Graph Fourier transform

- Example on a simple graph

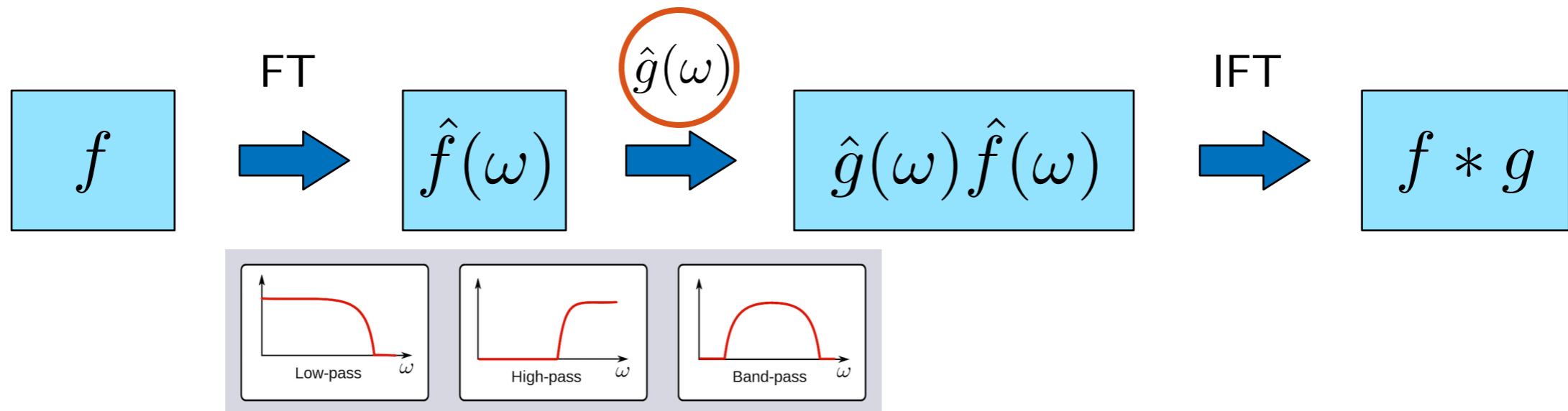


# Outline

- Graph signal processing (GSP): Basic concepts
- Graph spectral filtering: Basic tools of GSP
- Connection with literature
- Representation of graph signals
- Applications

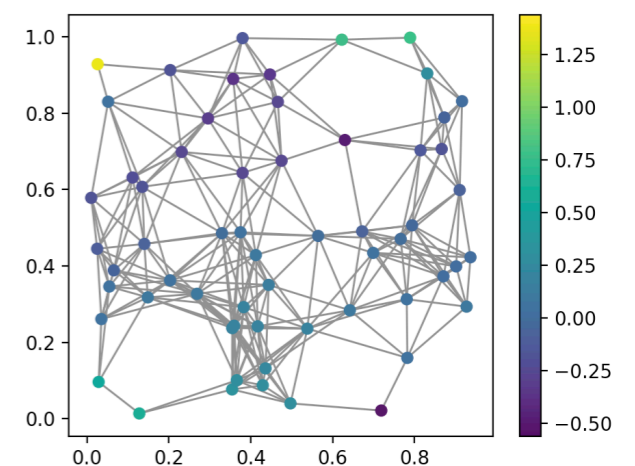
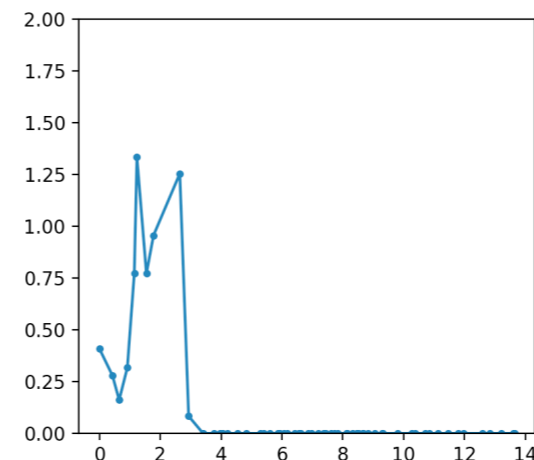
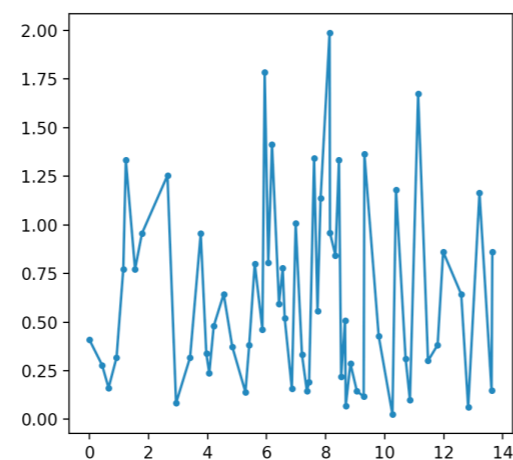
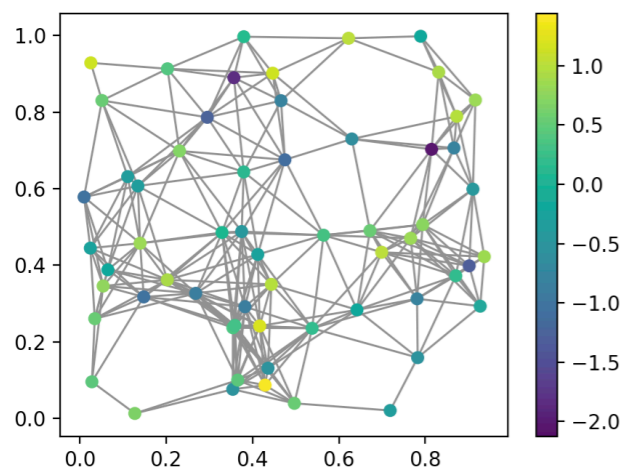
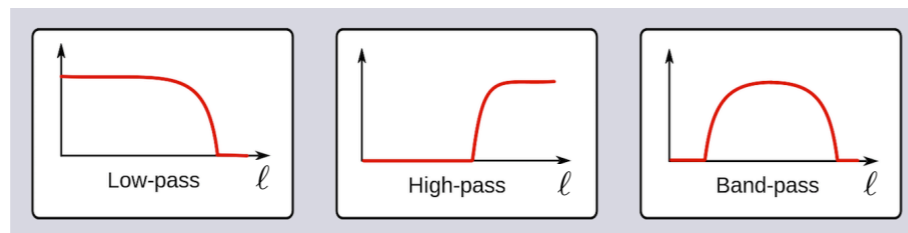
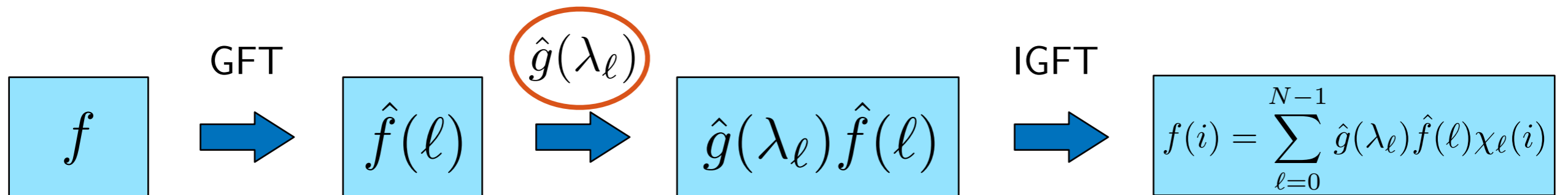
# Classical frequency filtering

Classical FT:  $\hat{f}(\omega) = \int (e^{j\omega x})^* f(x) dx$      $f(x) = \frac{1}{2\pi} \int \hat{f}(\omega) e^{j\omega x} d\omega$



# Graph spectral filtering

$$\text{GFT: } \hat{f}(\ell) = \langle \chi_\ell, f \rangle = \sum_{i=1}^N \chi_\ell^*(i) f(i) \quad f(i) = \sum_{\ell=0}^{N-1} \hat{f}(\ell) \chi_\ell(i)$$

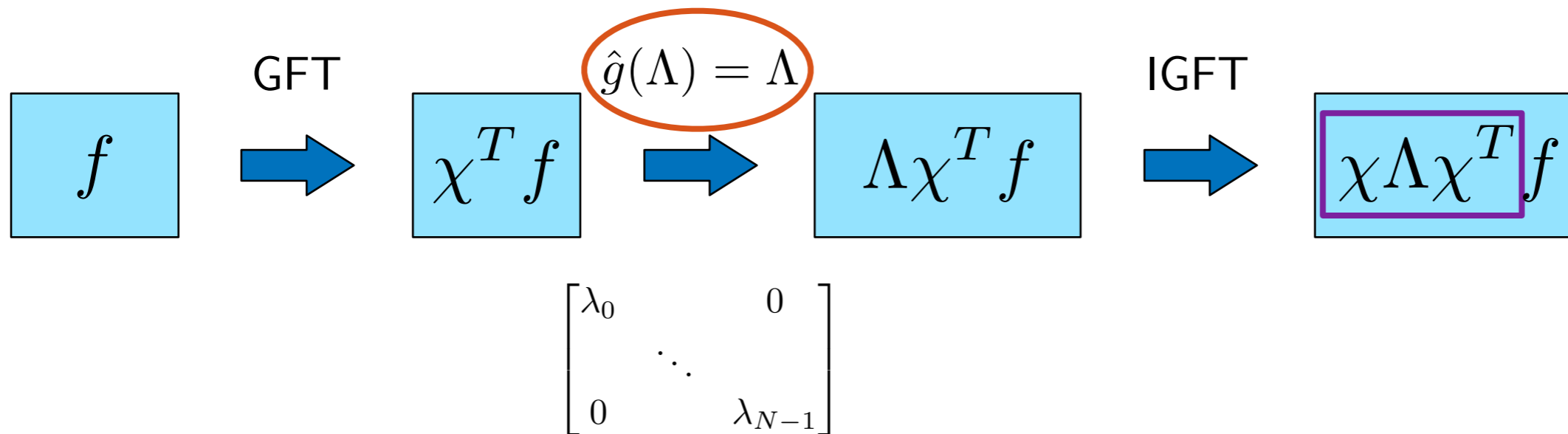




# Graph Laplacian revisited

$$\text{GFT: } \hat{f}(\ell) = \langle \chi_\ell, f \rangle = \sum_{i=1}^N \chi_\ell^*(i) f(i) \quad f(i) = \sum_{\ell=0}^{N-1} \hat{f}(\ell) \chi_\ell(i)$$

The Laplacian  $L$  is a difference operator:  $Lf = \chi \Lambda \chi^T f$

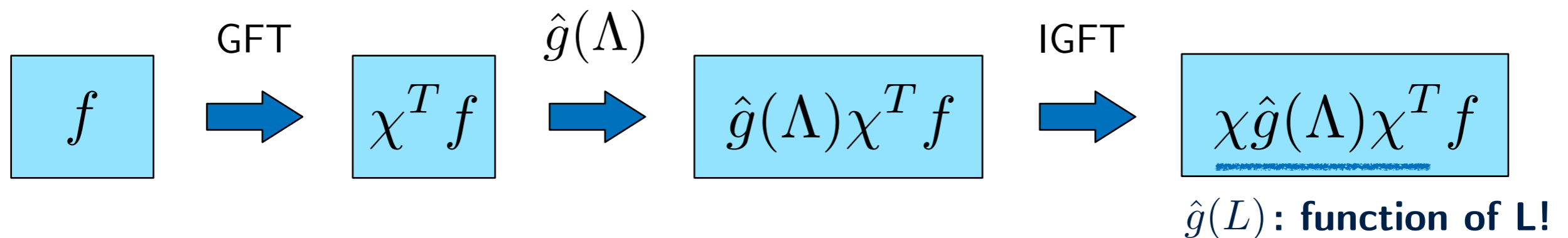


The Laplacian operator filters the signal in the spectral domain by its eigenvalues!

The Laplacian quadratic form:  $f^T L f = \|L^{\frac{1}{2}} f\|_2^2 = \|\chi \Lambda^{\frac{1}{2}} \chi^T f\|_2^2$

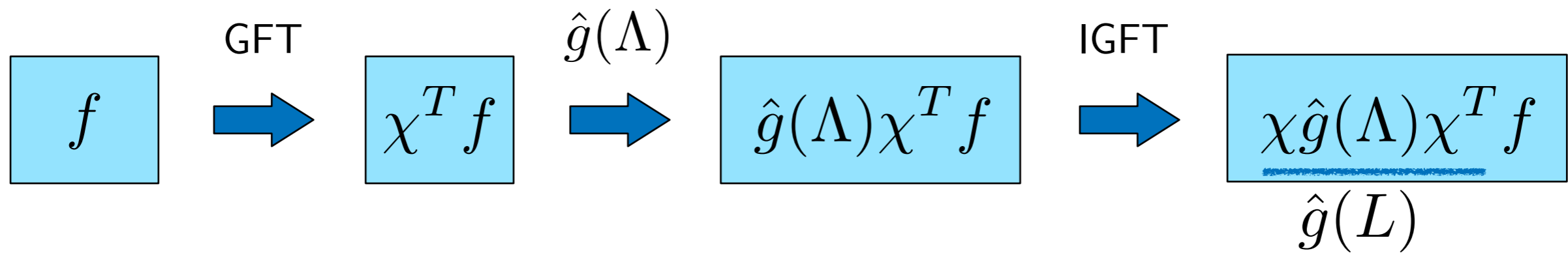
# Graph transform/dictionary design

- Transforms and dictionaries can be designed through graph spectral filtering: Functions of graph Laplacian!



- Important properties can be achieved by properly defining  $\hat{g}(L)$ , such as localisation of atoms
- Closely related to kernels and regularisation on graphs

# A practical example



problem: we observe a noisy graph signal  $f = y_0 + \eta$  and wish to recover  $y_0$

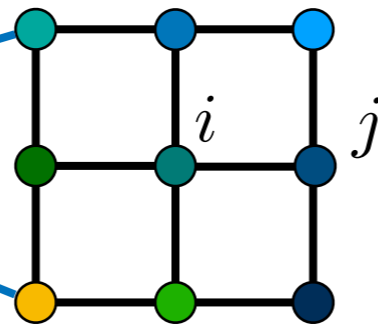
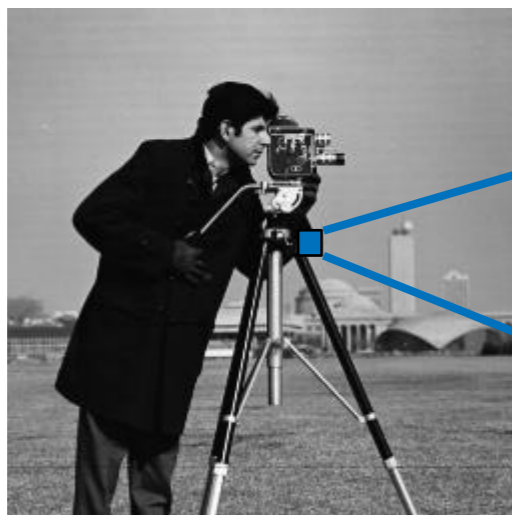
$$y^* = \arg \min_y \{ \underbrace{\|y - f\|_2^2}_{\text{data fitting term}} + \underbrace{\gamma y^T L y}_{\text{"smoothness" assumption}} \}$$

$$y^* = \underbrace{(I + \gamma L)^{-1} f}_{\hat{g}(L)} = \chi(1 + \gamma\Lambda)^{-1}\chi^T f$$

**remove noise by low-pass filtering  
in graph spectral domain!**

# A practical example

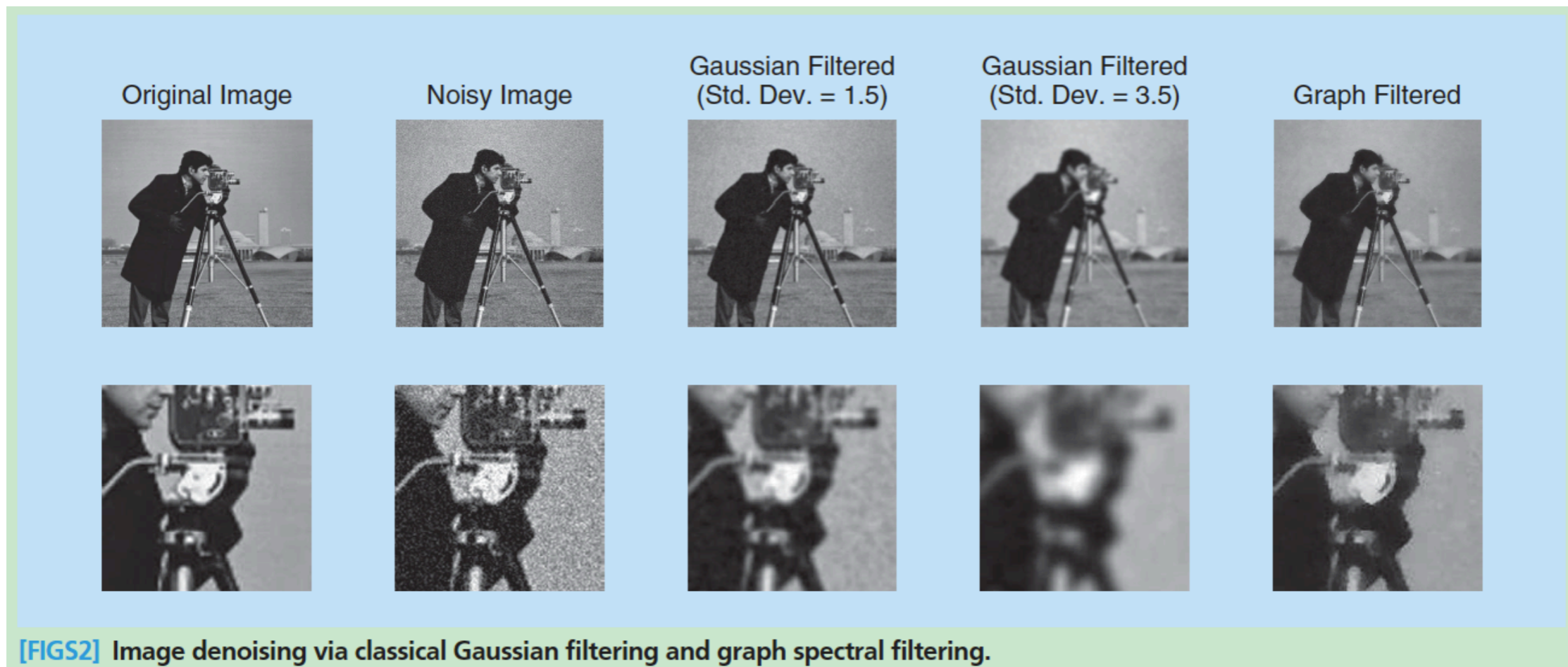
- noisy image as observed noisy graph signal
- regular grid graph (weights inversely proportional to pixel value difference)



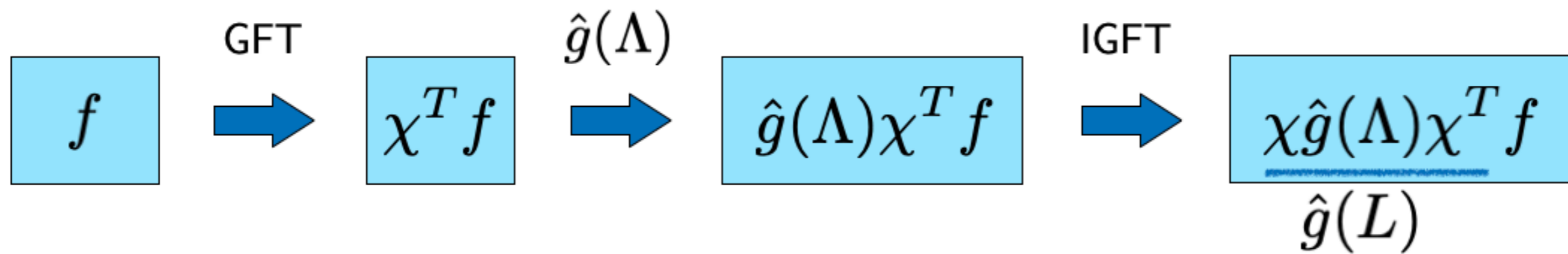
$$w_{ij} = \frac{1}{|f(i) - f(j)|}$$

# A practical example

- noisy image as observed noisy graph signal
- regular grid graph (weights inversely proportional to pixel value difference)



# Graph transform/dictionary design



smoothing/low-pass filtering:  $\hat{g}(L) = (I + \gamma L)^{-1} = \chi(I + \gamma\Lambda)^{-1}\chi^T$

**Graph-based  
regularisation**

windowed kernel: windowed graph Fourier transform

shifted and dilated band-pass filters: spectral graph wavelets  $\hat{g}(sL)$

**Graph filters  
& transforms**

adapted kernels: learn values of  $\hat{g}(L)$  directly from data

parametric kernel:  $\hat{g}(L) = \sum_{k=0}^K \theta_j L^k = \chi\left(\sum_{k=0}^K \theta_j \Lambda^k\right)\chi^T$

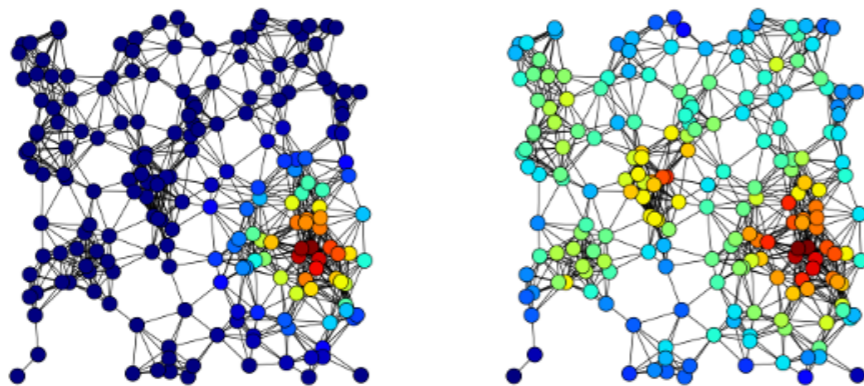
**Learning models  
on graphs**

# Outline

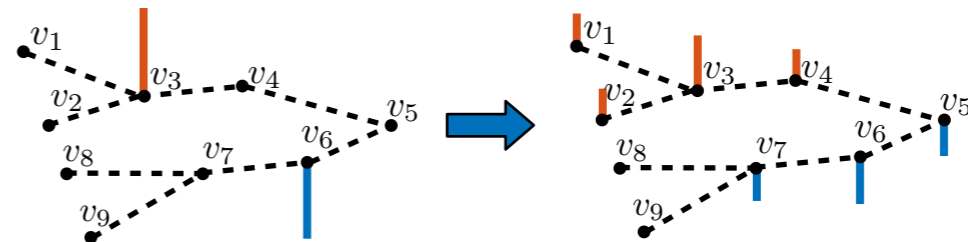
- Graph signal processing (GSP): Basic concepts
- Graph spectral filtering: Basic tools of GSP
- Connection with literature
- Representation of graph signals
- Applications

# GSP and the literature

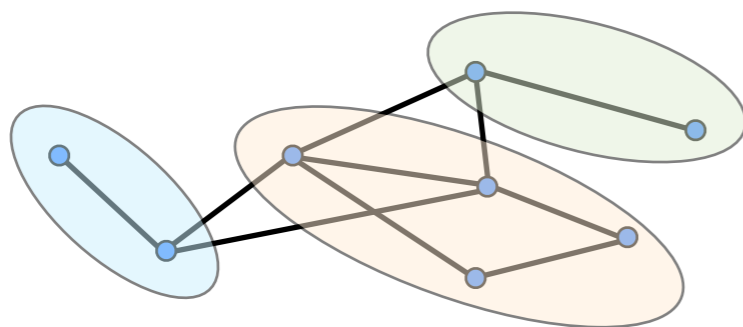
there is a rich literature about data analysis and learning on graphs



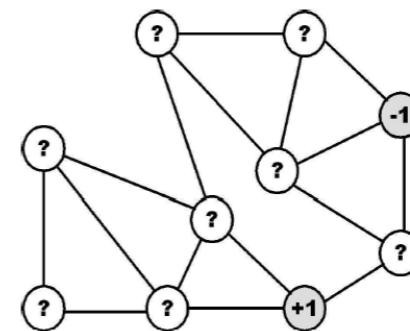
network science (node centrality)



diffusion on graphs



unsupervised learning (dimensionality reduction, clustering)

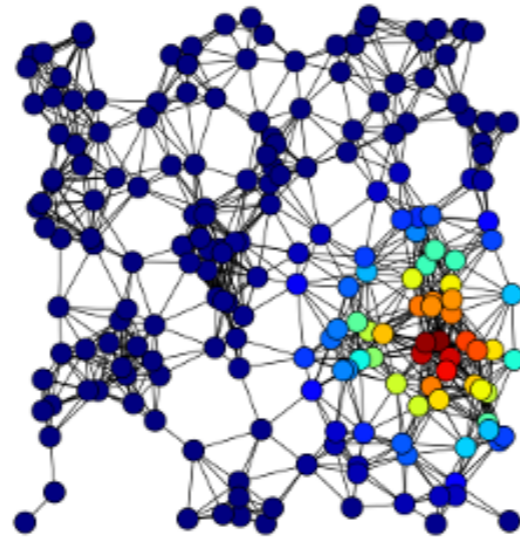


semi-supervised learning



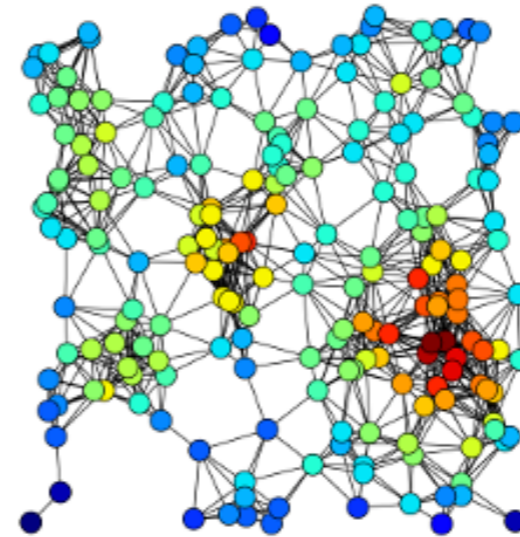
# Network centrality

eigenvector centrality



$$Wx = \lambda_{\max}x$$

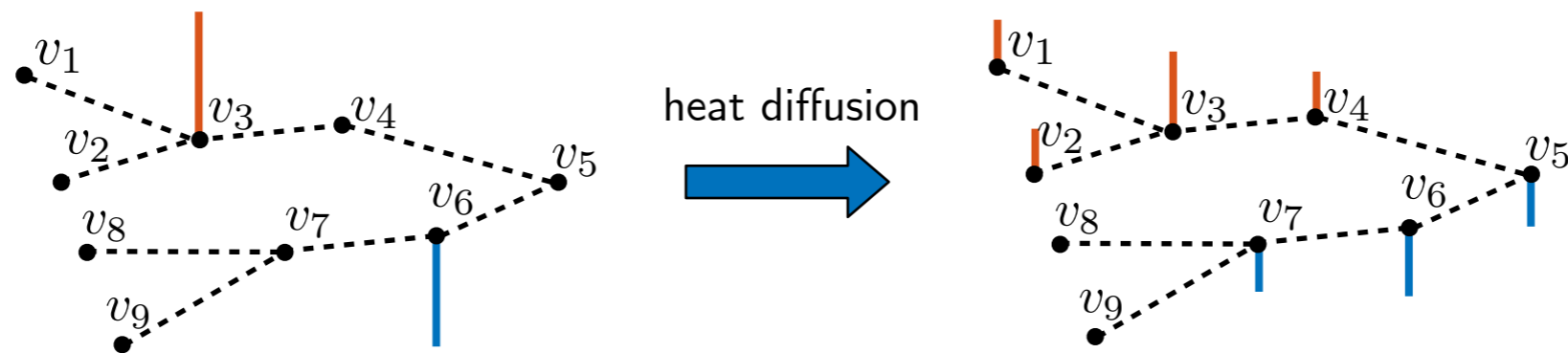
degree centrality



$$d = [d(v_1), \dots, d(v_N)]$$

- Google's PageRank is a variant of eigenvector centrality
- eigenvectors of  $W$  can also be used to provide a frequency interpretation for graph signals

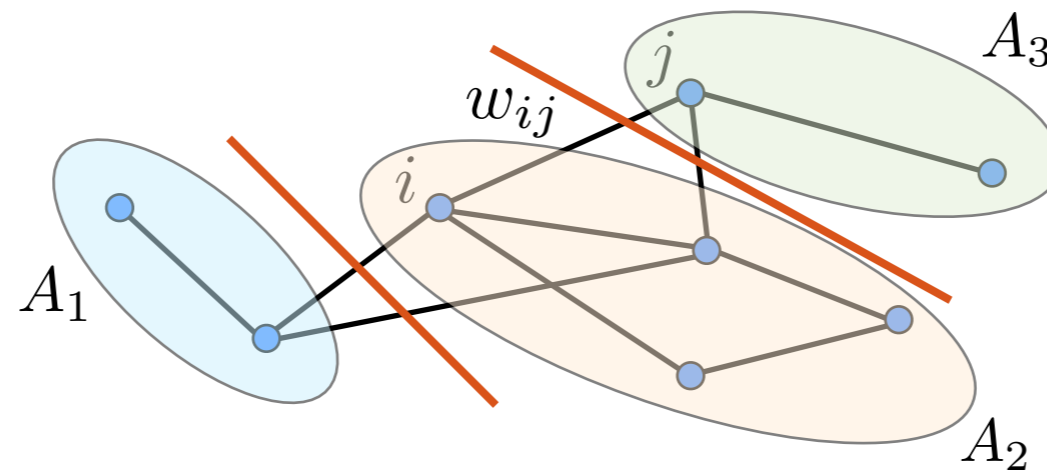
# Diffusion on graphs



$$\begin{aligned} \frac{\partial x}{\partial \tau} + Lx &= 0 \\ x(v, 0) &= x_0(v) \end{aligned} \quad \longrightarrow \quad x(v, \tau) = e^{-\tau L} x_0(v)$$

- heat diffusion on graphs is a typical physical process on graphs
- other possibilities exist (e.g., random walk on graphs)
- many have an interpretation of filtering on graphs

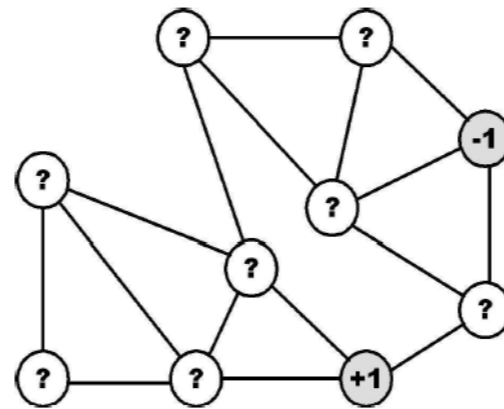
# Graph clustering (community detection)



$$NCut(A_1, \dots, A_k) = \frac{1}{2} \sum_{i=1}^k \frac{W(A_i, \bar{A}_i)}{vol(A_i)}$$

- first  $k$  eigenvectors of graph Laplacian provide solution to graph cut minimisation
- eigenvectors of graph Laplacian enable a Fourier-like analysis for graph signals

# Semi-supervised learning



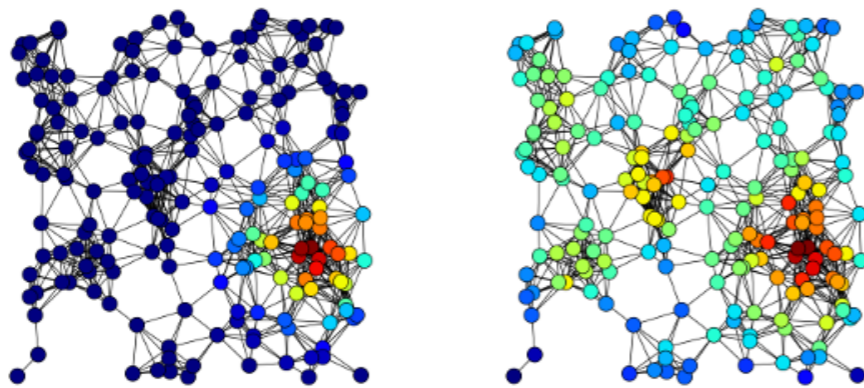
$$y : \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ +1 \\ 0 \\ 0 \\ -1 \\ 0 \end{bmatrix}$$

$$\min_{x \in \mathbb{R}^N} \|y - x\|_2^2 + \alpha x^T L x,$$

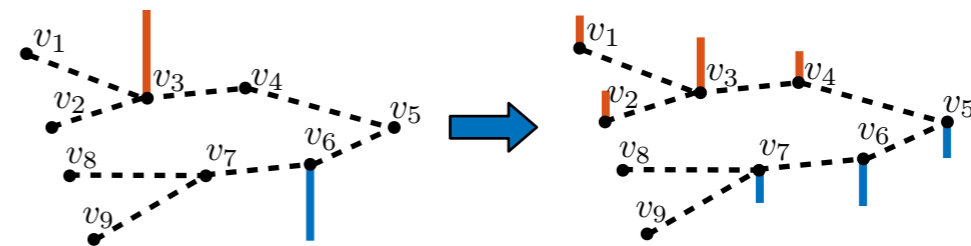
- learning by assuming smoothness of predicted labels (label propagation)
- this is equivalent to a denoising problem for graph signal  $y$

# GSP and the literature

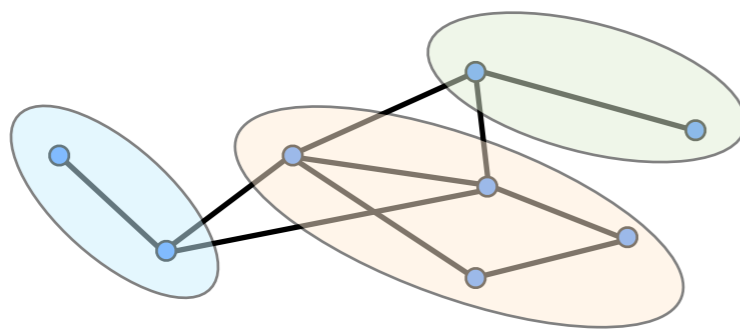
centrality, diffused information, cluster membership, node labels  
(and node features in general) can ALL be viewed as graph signals



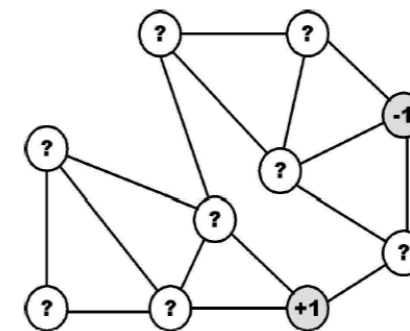
network science



diffusion on graphs



unsupervised learning (dimensionality reduction, clustering)



semi-supervised learning

# Outline

- Graph signal processing (GSP): Basic concepts
- Graph spectral filtering: Basic tools of GSP
- Connection with literature
- Representation of graph signals
- Applications

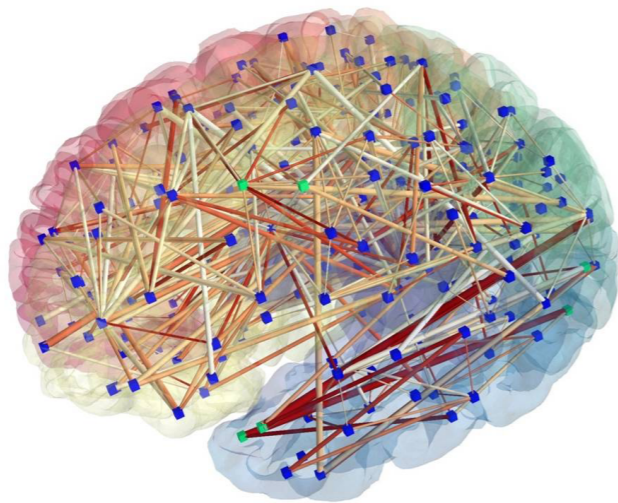
# Why representations for graph signals?



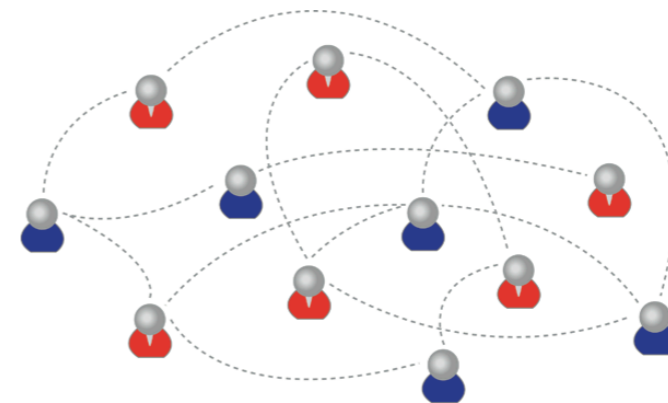
**image analysis (e.g.,  
denoising, compression)**



**traffic analysis (e.g.,  
mobility, congestion)**



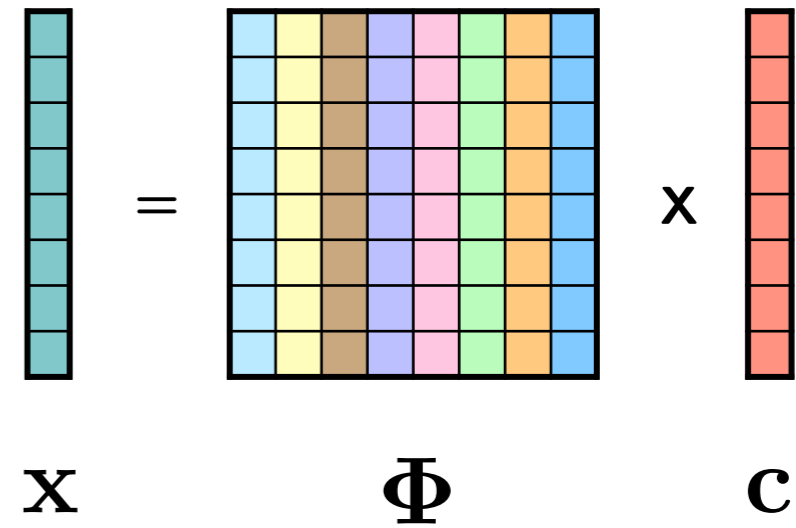
**neuroscience (e.g.,  
brain analysis)**



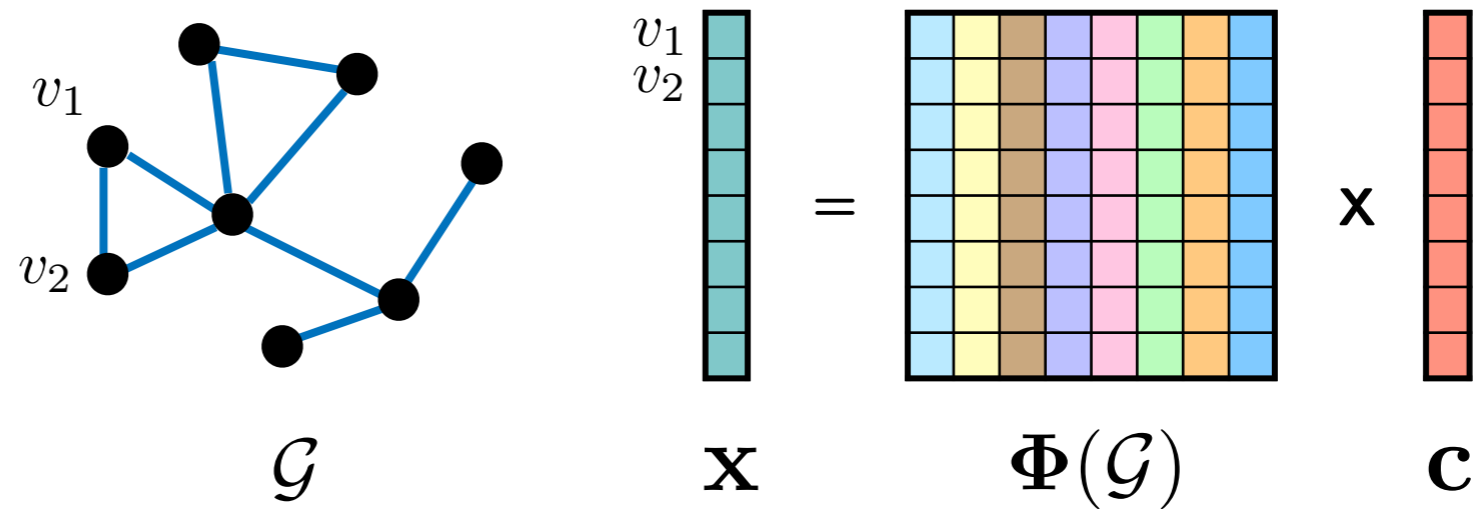
**social network analysis (e.g.,  
community, recommendation)**

# Classical vs. Graph dictionaries

classical signal

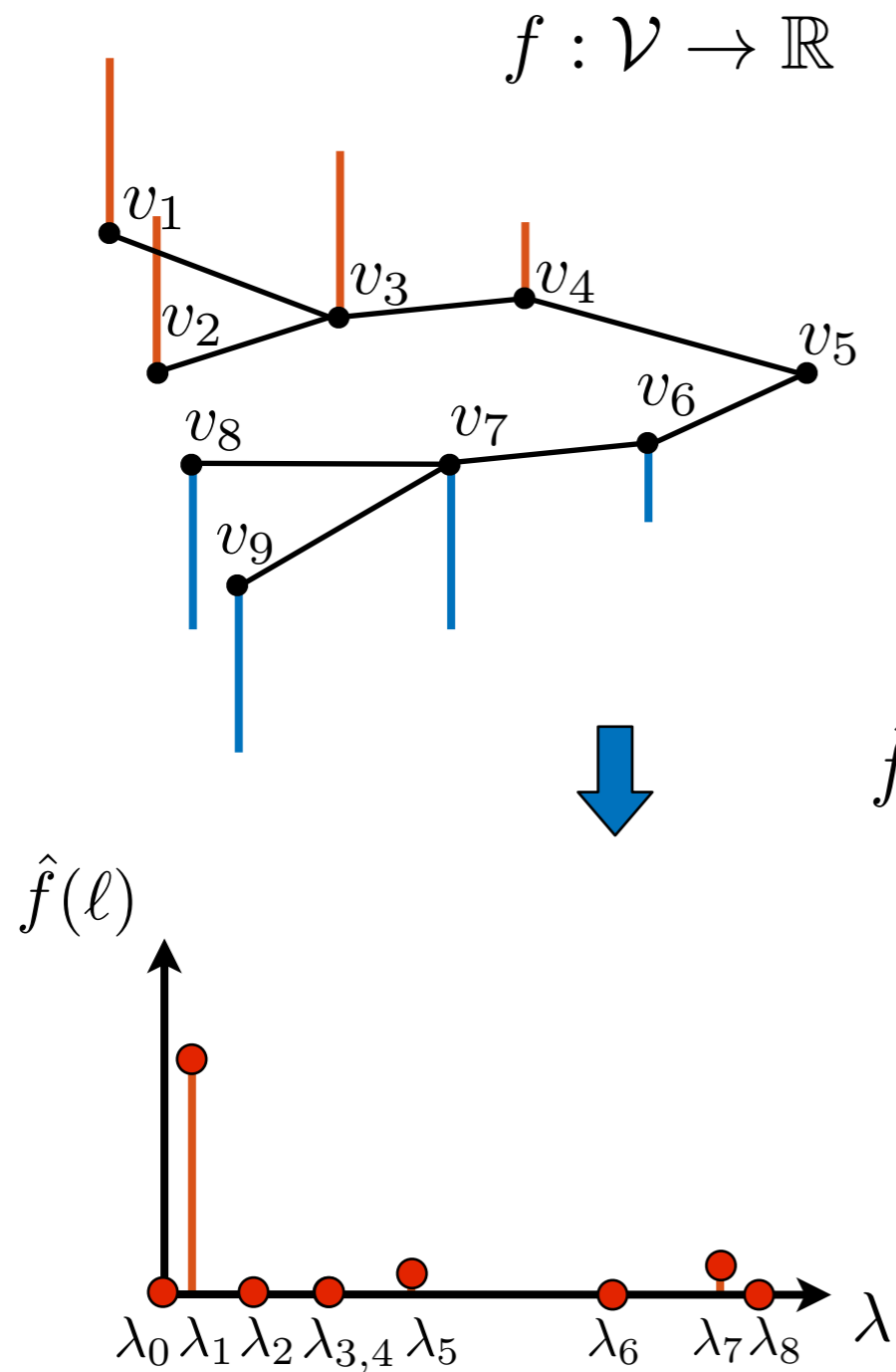


graph signal





# GFT as a first graph dictionary

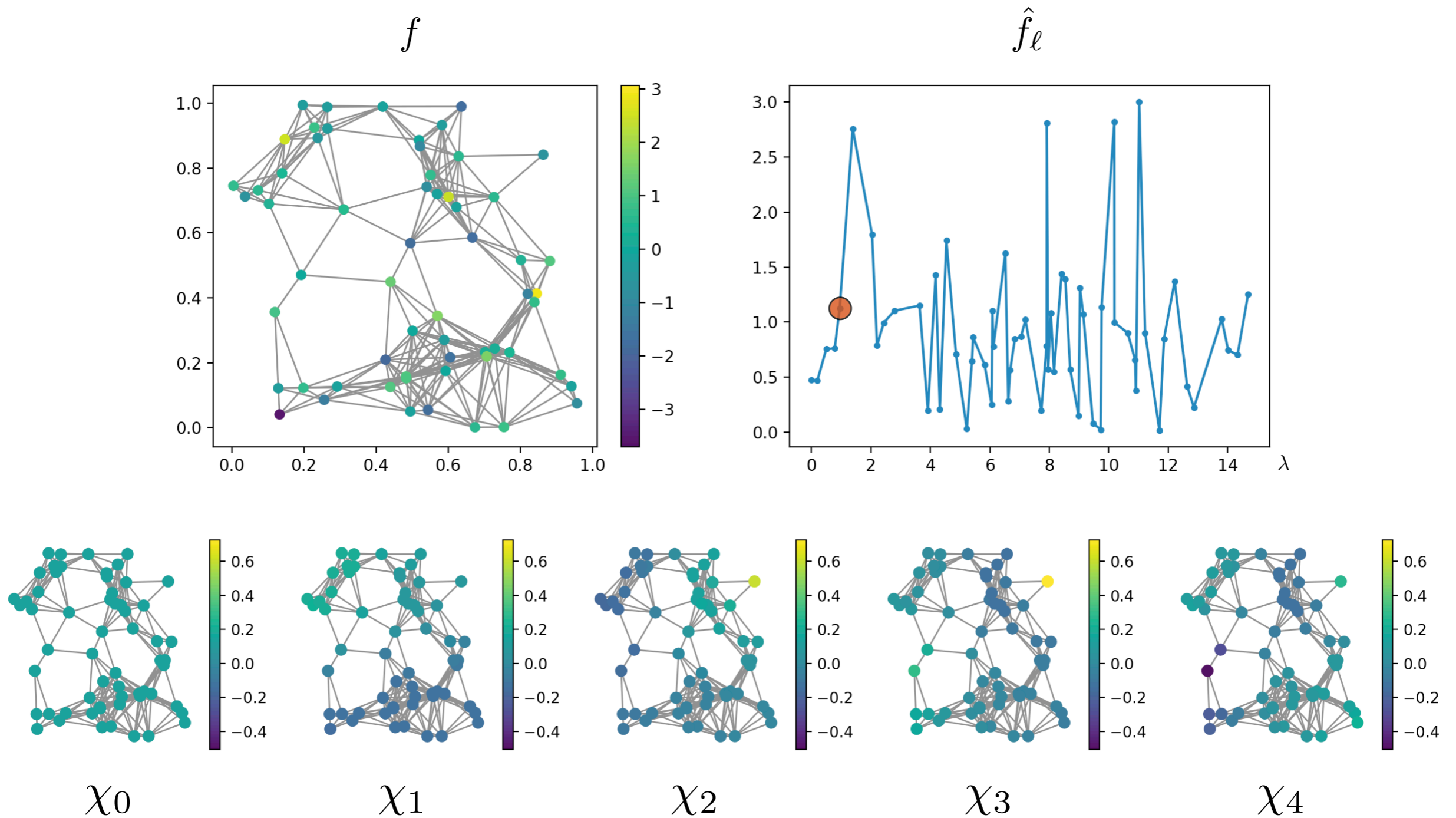


vertex domain

$$\hat{f}(\ell) = \langle \chi_\ell, f \rangle : \begin{bmatrix} | & & | \\ \chi_0 & \cdots & \chi_{N-1} \\ | & & | \end{bmatrix}^T \begin{matrix} | \\ f \\ | \end{matrix}$$

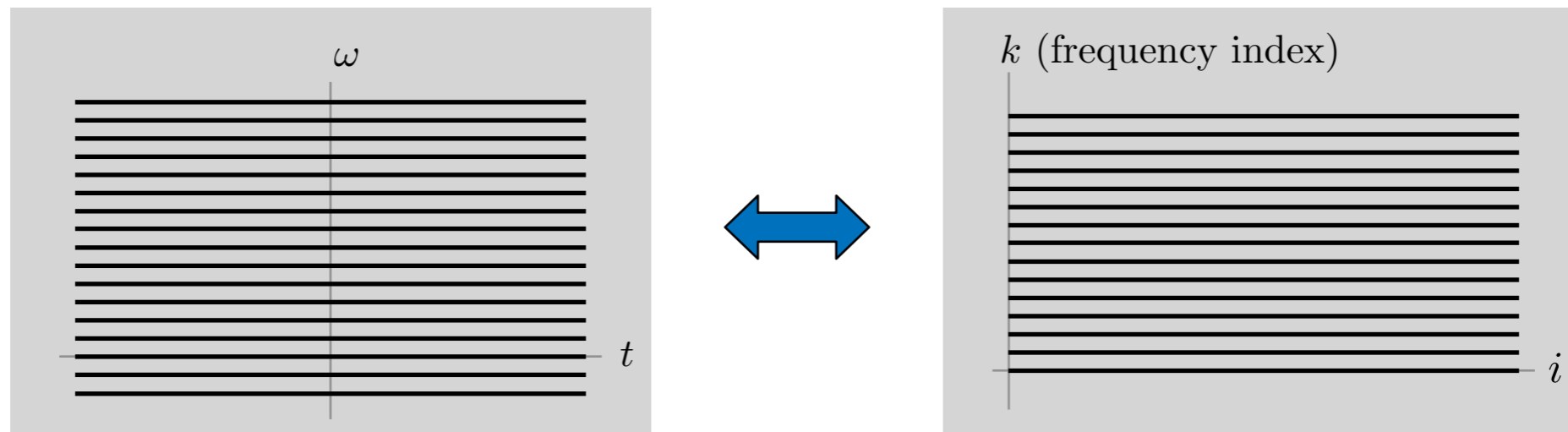
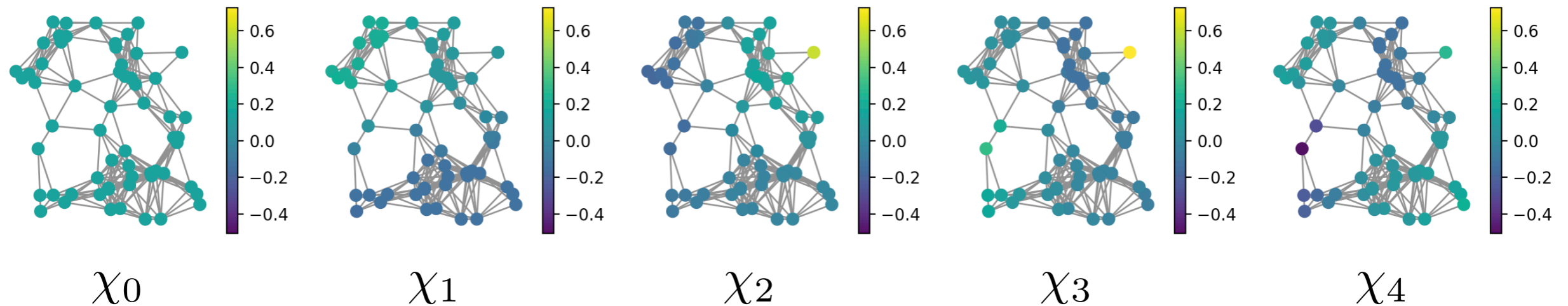
frequency (graph spectral) domain

# GFT as a first graph dictionary



**GFT atoms (corresponding to discrete frequencies)**

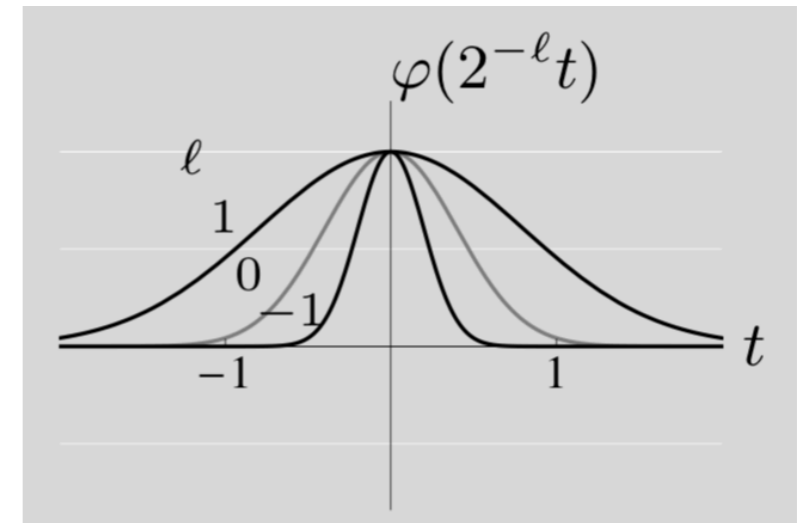
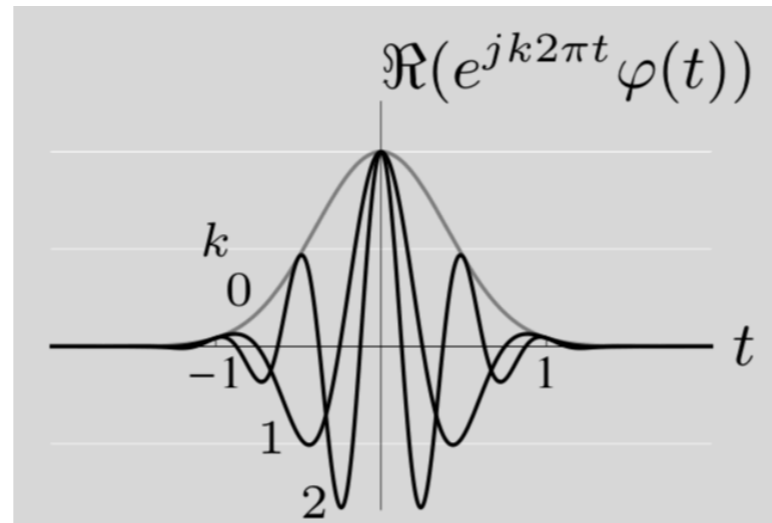
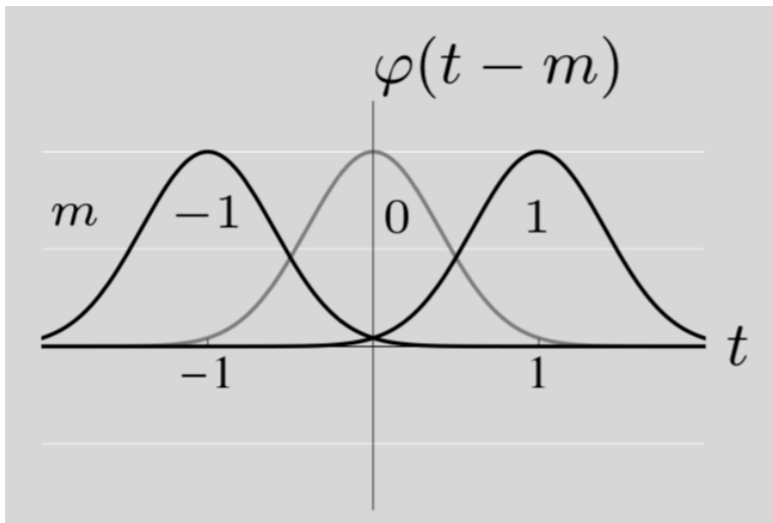
# GFT as a first graph dictionary



- like complex exponentials in classical FT, eigenvectors in GFT have **global** support
- can we design **localised** atoms on graphs?

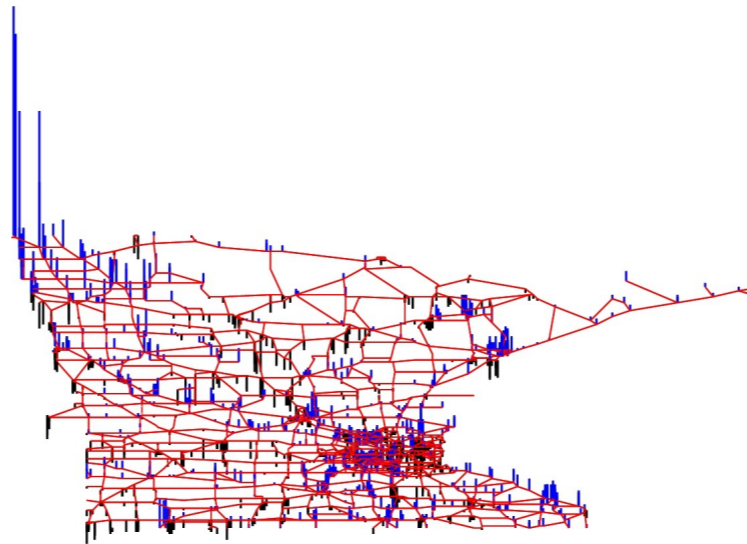
# Basic operations for graph signals

## basic operations in Euclidean domain



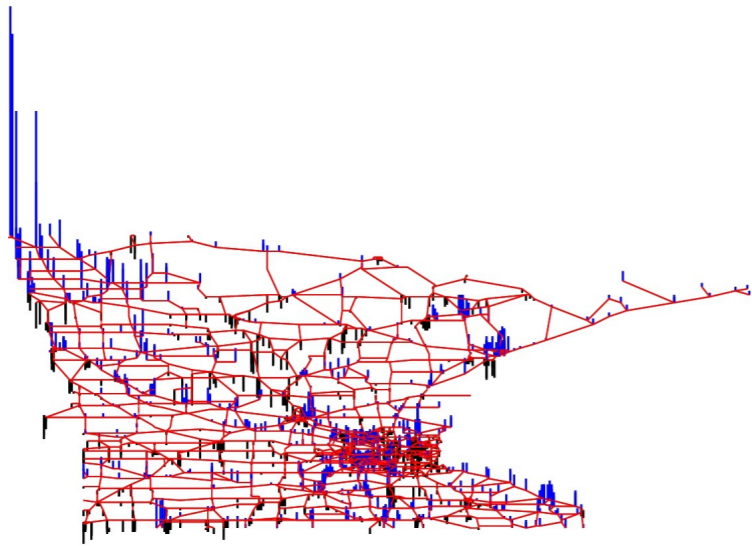
- recall that we used a set of structured functions (e.g., shifted and modulated) to produce localised items

# Basic operations for graph signals



- recall that we used a set of structured functions (e.g., shifted and modulated) to produce localised items
- we need to define for graph signals the basic operations of **convolution**, **shift**, **modulation**

# Convolution



**classical convolution**

$$(f * g)(t) = \int_{-\infty}^{\infty} f(t - \tau)g(\tau)d\tau$$

$$\widehat{(f * g)}(\omega) = \hat{f}(\omega) \cdot \hat{g}(\omega)$$

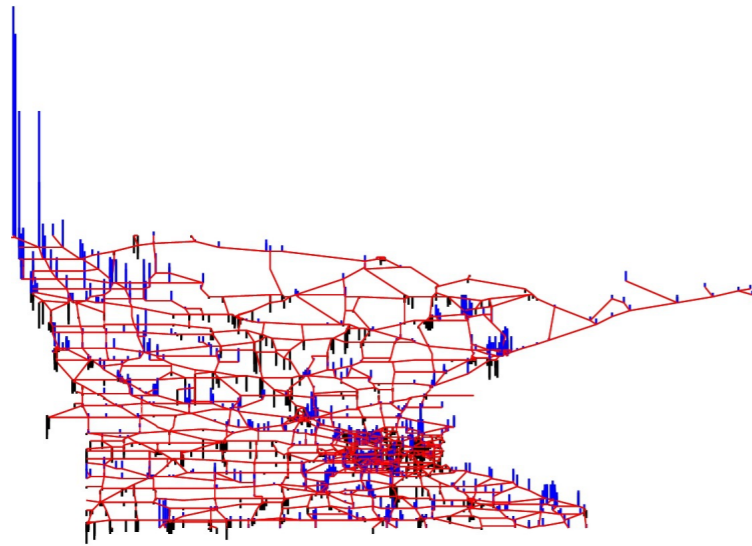
**graph convolution**

**multiplication in graph spectral domain**

$$\widehat{(f * g)}(\lambda) = (\hat{f} \circ \hat{g})(\lambda)$$

➔ 
$$(f * g)(n) := \sum_{\ell=0}^{N-1} \hat{f}(\ell)\hat{g}(\ell)\chi_{\ell}(n)$$

# Vertex-domain shift



original signal

**classical shift**

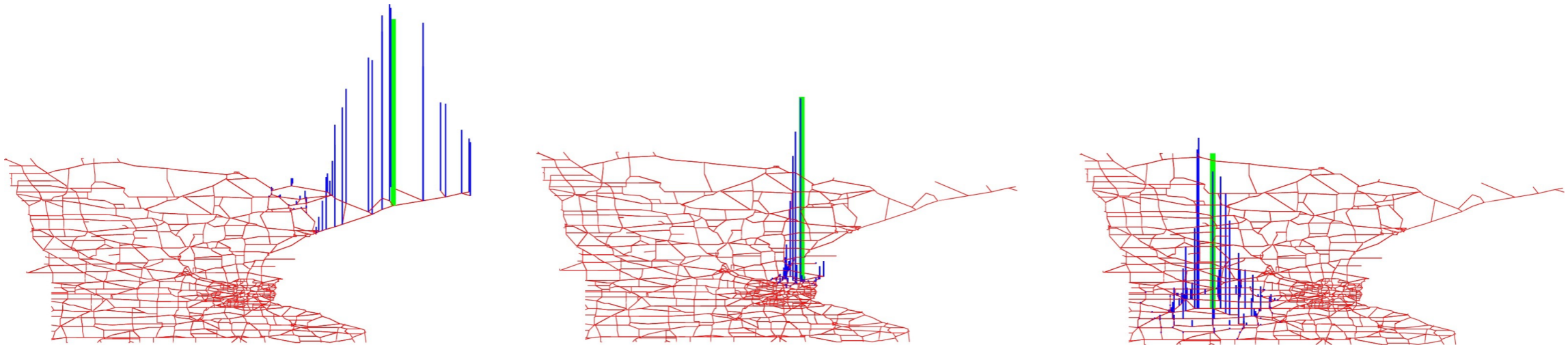
$$(T_u f)(t) := f(t - u) = (f * \delta_u)(t)$$

**graph shift**

**convolution with a “delta” on graph**

$$\begin{aligned} (T_i f)(n) &:= \sqrt{N} (f * \delta_i)(n) \\ &= \sqrt{N} \sum_{\ell=0}^{N-1} \hat{f}(\ell) \chi_{\ell}^*(i) \chi_{\ell}(n) \end{aligned}$$

# Vertex-domain shift



shifted version of the signal to different centring vertex (in green)

**classical shift**

$$(T_u f)(t) := f(t - u) = (f * \delta_u)(t)$$

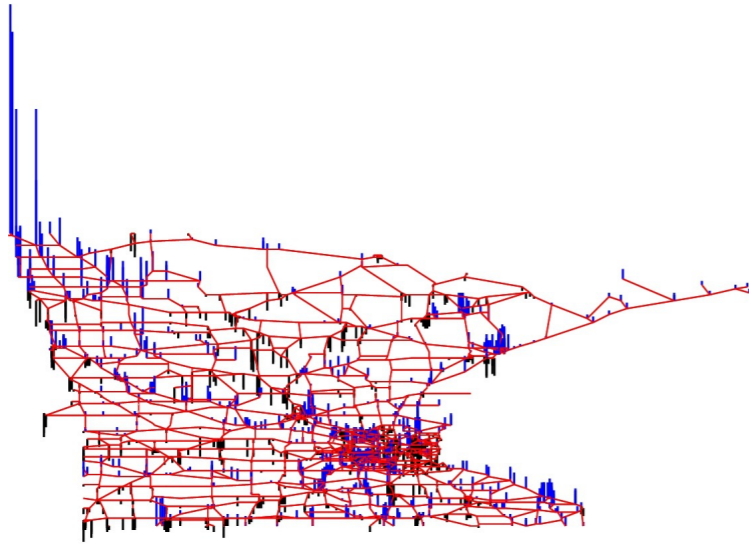
**graph shift**

**convolution with a “delta” on graph**

$$\begin{aligned} (T_i f)(n) &:= \sqrt{N} (f * \delta_i)(n) \\ &= \sqrt{N} \sum_{\ell=0}^{N-1} \hat{f}(\ell) \chi_{\ell}^*(i) \chi_{\ell}(n) \end{aligned}$$



# Modulation



**classical modulation**

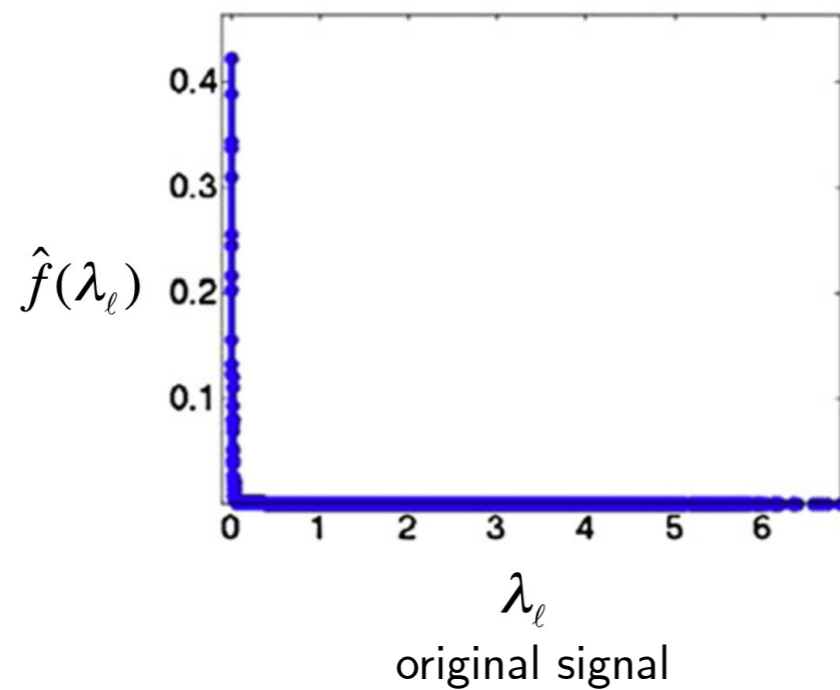
$$(M_{\xi} f)(t) := e^{j2\pi\xi t} f(t)$$

**graph modulation**

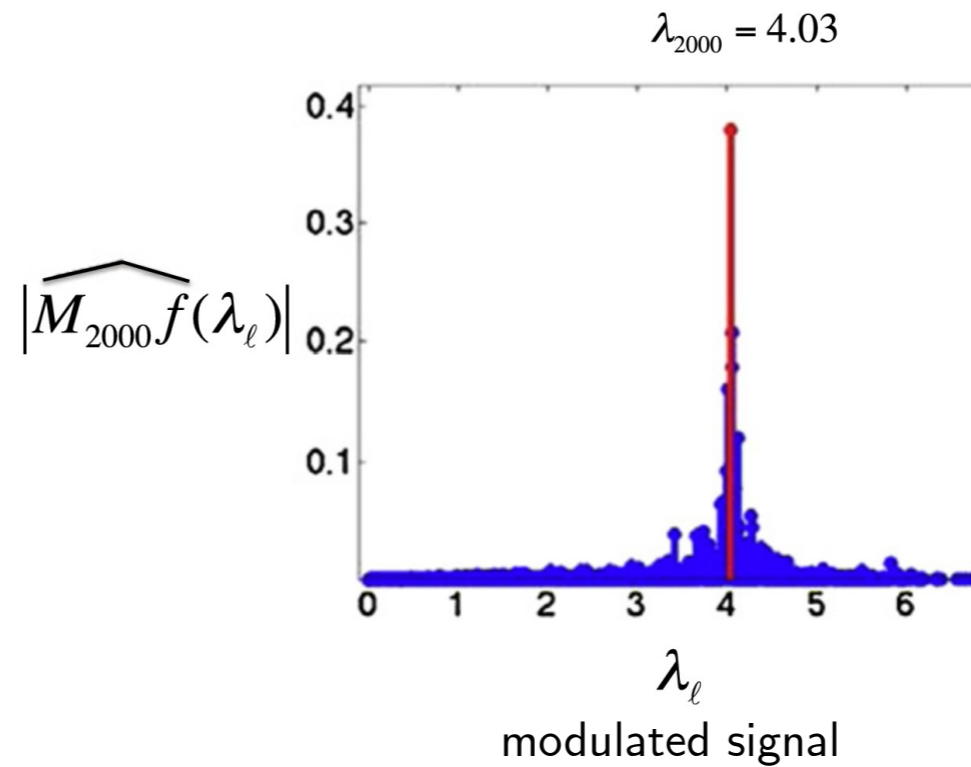
**multiply by a graph Laplacian eigenvector**

$$(M_k f)(n) := \sqrt{N} f(n) \chi_k(n)$$

# Modulation



**classical modulation**



$$(M_\xi f)(t) := e^{j2\pi\xi t} f(t)$$

**graph modulation**

**multiply by a graph Laplacian eigenvector**

$$(M_k f)(n) := \sqrt{N} f(n) \chi_k(n)$$

# Windowed graph Fourier transform

- With the shift and modulation operators for graph signals we can now define a windowed graph Fourier transform

**classical windowed  
Fourier atom**

$$g_{u,\xi}(t) := (M_\xi T_u g)(t) = e^{j2\pi\xi t} g(t-u)$$

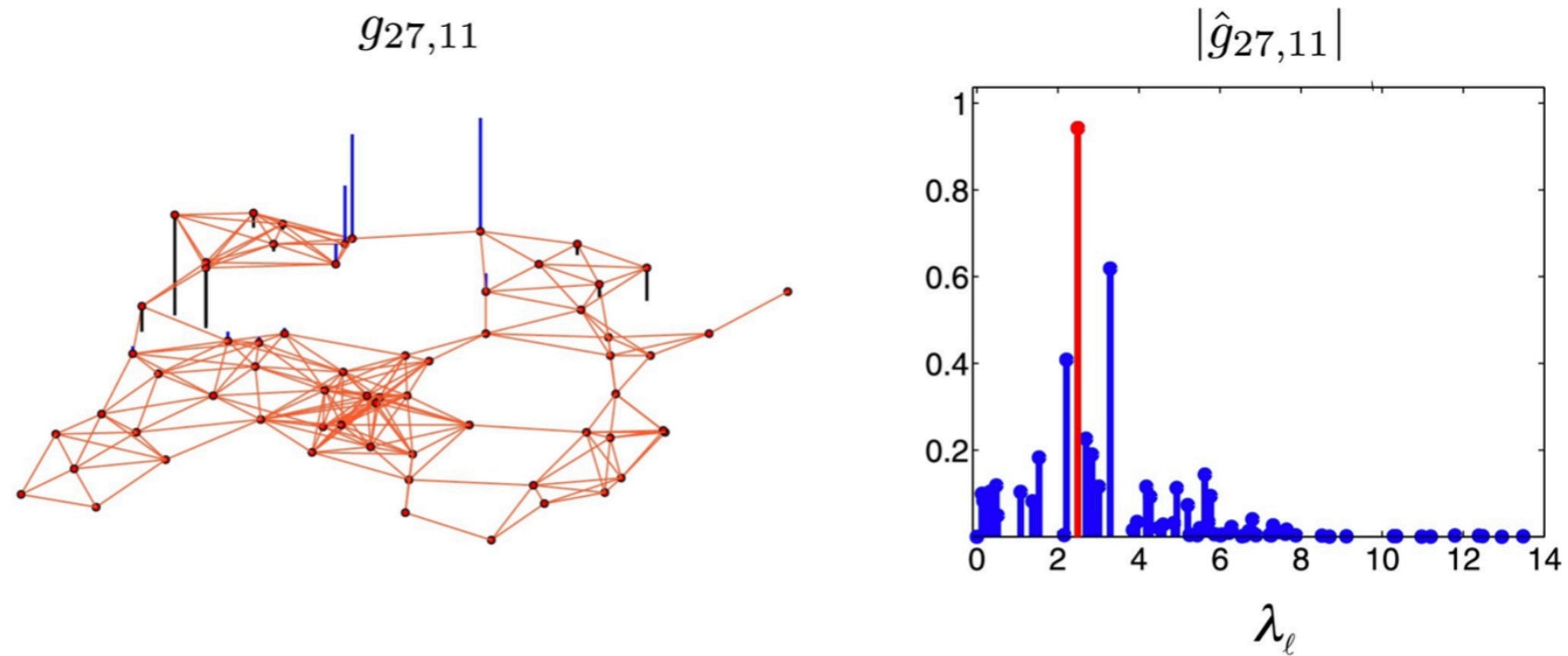
**windowed graph  
Fourier atom**

$$g_{i,k}(n) := (M_k T_i g)(n) \\ = N \chi_k(n) \sum_{\ell=0}^{N-1} \hat{g}(\lambda_\ell) \chi_\ell^*(i) \chi_\ell(n)$$

**windowed graph Fourier  
transform (WGFT)**

$$Sf(i, k) := \langle f, g_{i,k} \rangle$$

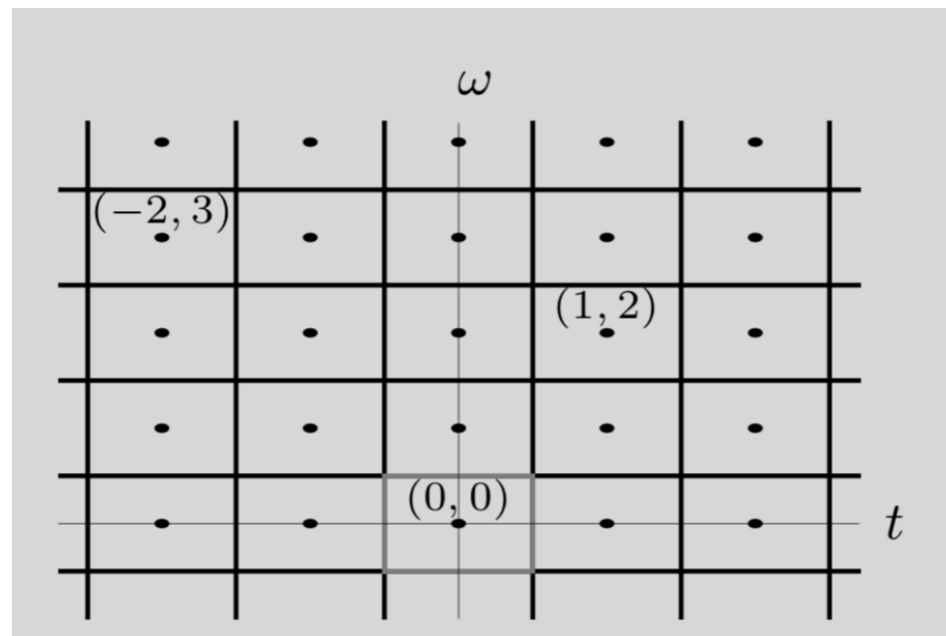
# Windowed graph Fourier transform



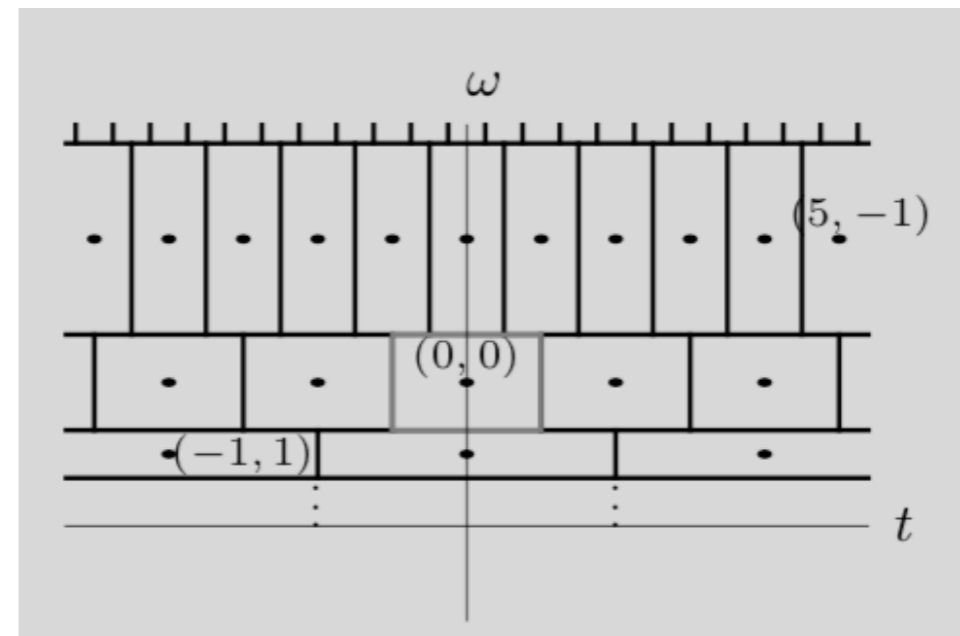
$$\hat{g}(\lambda_\ell) = 1.45e^{-3\lambda_\ell}$$

# Fourier vs wavelet atoms

**Fourier atoms**  
(time shift and modulation)



**wavelet atoms**  
(time shift and scaling)



$$\varphi_{k,m}(t) = e^{jk\omega_0 t} \varphi(t - mt_0) \quad k, m \in \mathbb{Z}$$

$$\varphi_{l,m}(t) = \varphi(a^{-l}t - mt_0) \quad l, m \in \mathbb{Z}$$

# Spectral graph wavelet transform

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



$$W_f(s, a) = \int_{-\infty}^{\infty} \frac{1}{s} \psi^*\left(\frac{x-a}{s}\right) f(x) dx$$



$$\bar{\psi}_s(x) = \frac{1}{s} \psi^*\left(\frac{-x}{s}\right)$$

$$\begin{aligned} (T^s f)(a) &= \int_{-\infty}^{\infty} \frac{1}{s} \psi^*\left(\frac{x-a}{s}\right) f(x) dx = \int_{-\infty}^{\infty} \bar{\psi}_s(a-x) f(x) dx \\ &= (\bar{\psi}_s \star f)(a) \end{aligned}$$



$$\widehat{T^s f}(\omega) = \hat{\psi}_s(\omega) \hat{f}(\omega) = \hat{\psi}^*(s\omega) \hat{f}(\omega)$$



$$(\widehat{T_g^s f})(\ell) = \hat{g}(s\lambda_\ell) \hat{f}(\ell)$$

$$(T^s f)(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{i\omega x} \hat{\psi}^*(s\omega) \hat{f}(\omega) d\omega$$

Fourier multiplier operator: scaled kernel  $\hat{\psi}^*(s\omega)$

**SGWT atom**

$$\psi_{s,i}(n) := (T_g^s \delta_i)(n) = \sum_{\ell=0}^{N-1} \hat{g}(s\lambda_\ell) \chi_\ell^*(i) \chi_\ell(n)$$

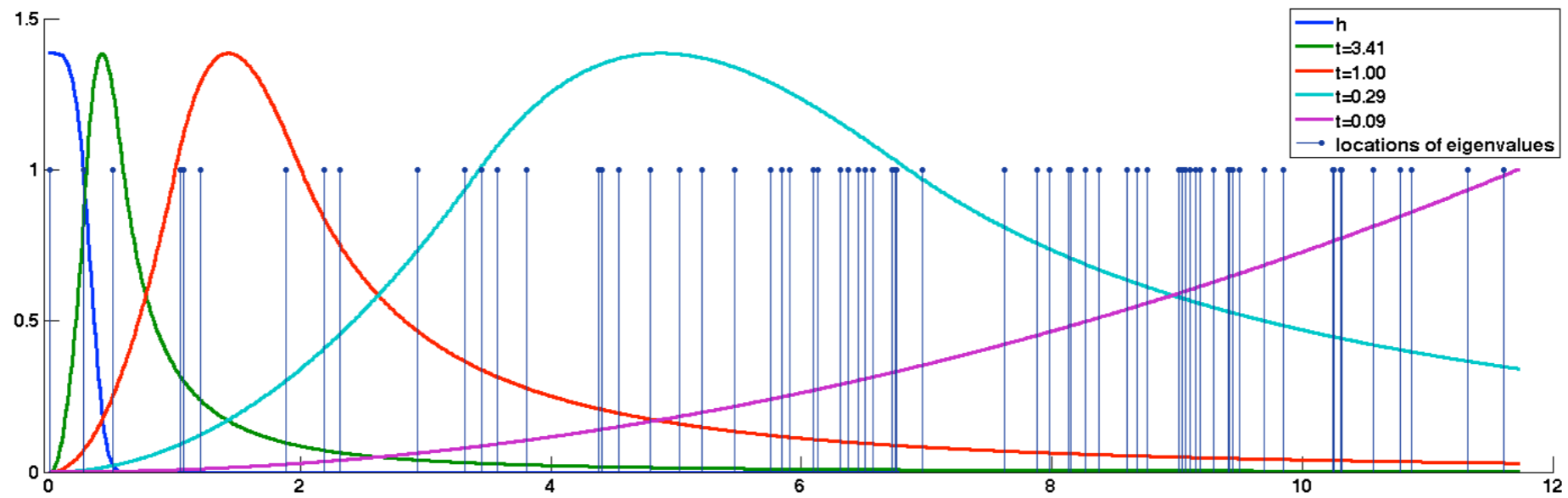


**spectral graph wavelet transform (SGWT)**

$$(T_g^s f)(n) = \sum_{\ell=0}^{N-1} \hat{g}(s\lambda_\ell) \hat{f}(\ell) \chi_\ell(n)$$



# Spectral graph wavelet transform



SGWT kernel functions  $\hat{g}(s\lambda_\ell)$

# Spectral graph wavelet transform

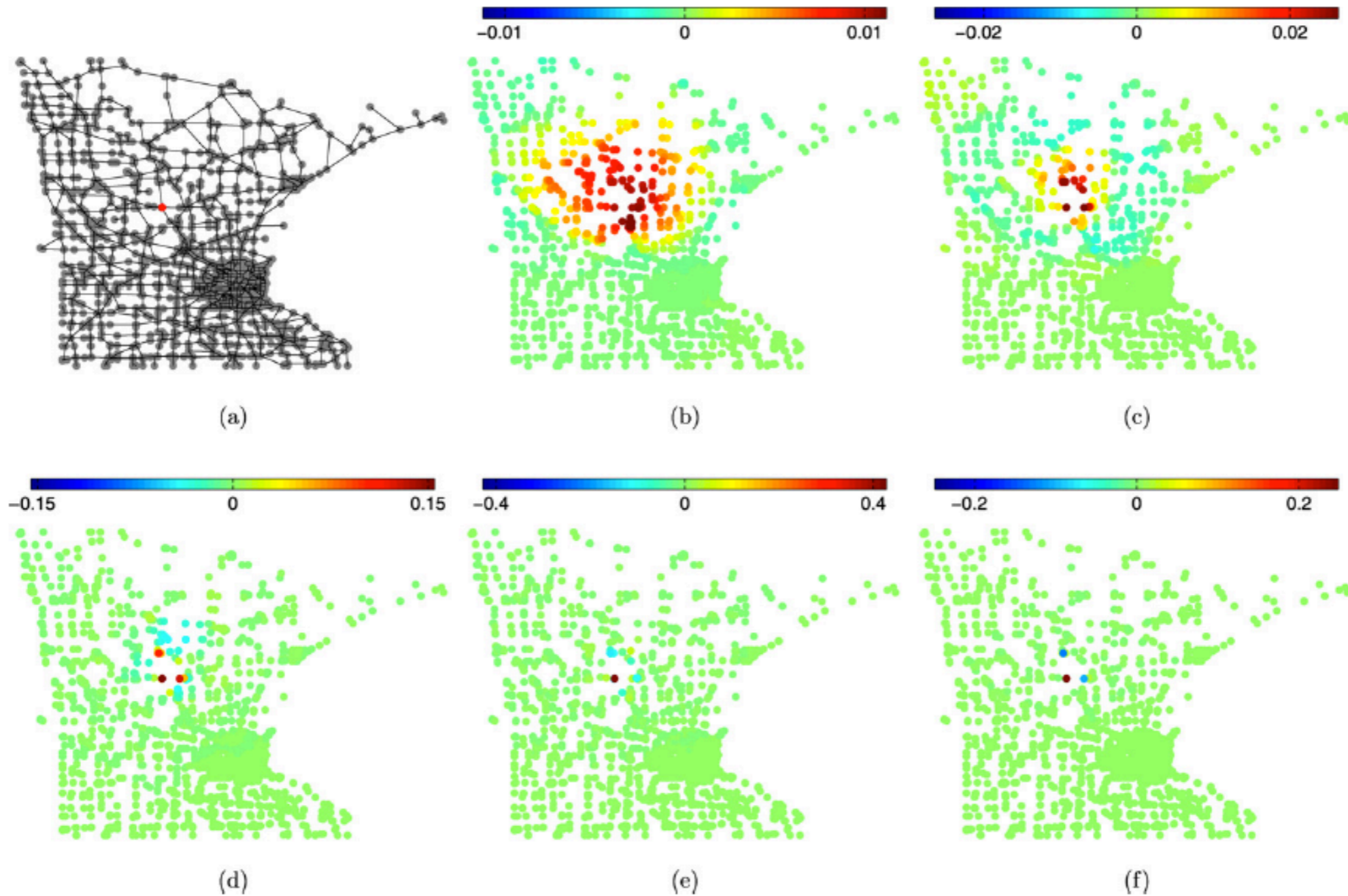
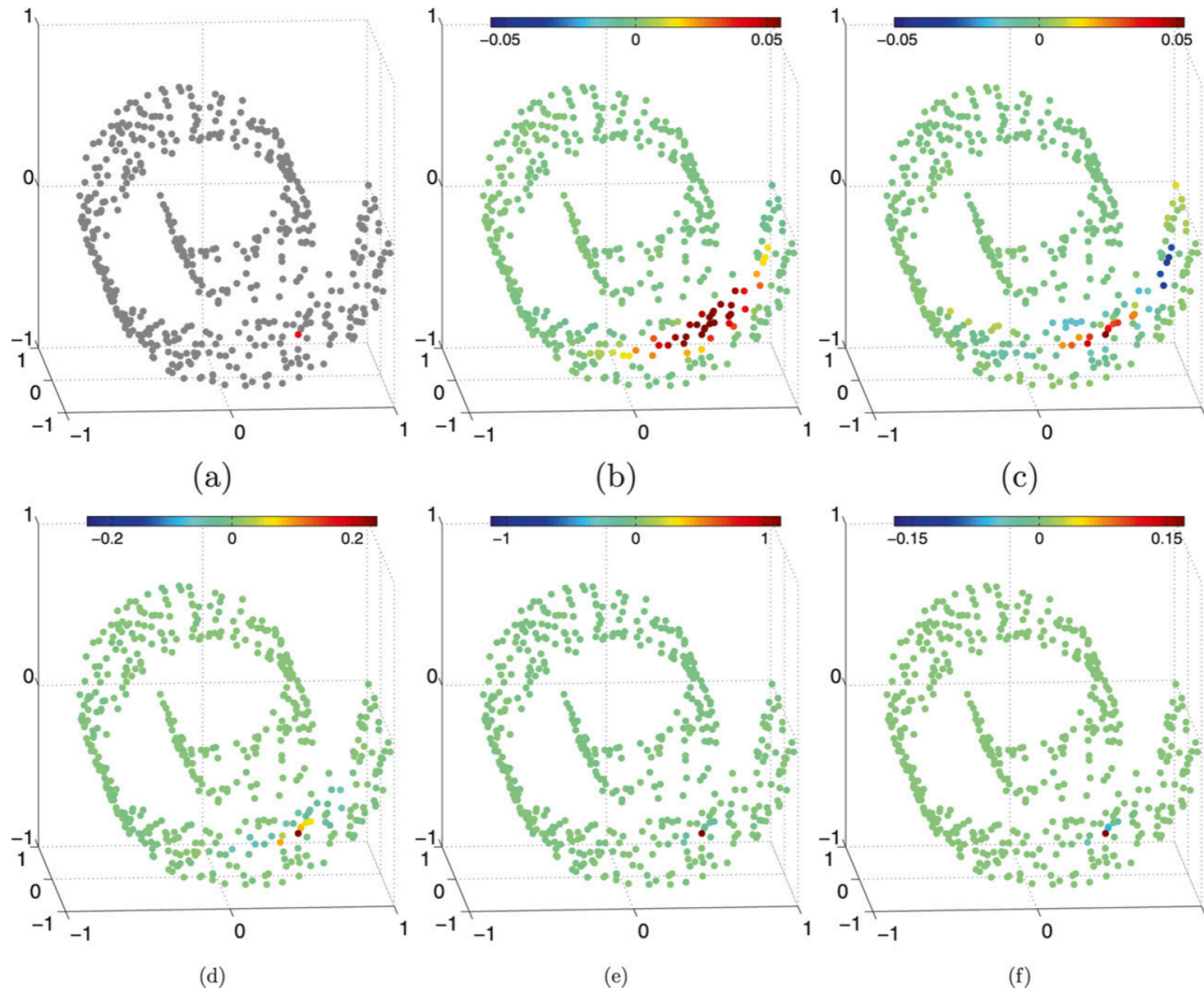


Fig. 4. Spectral graph wavelets on Minnesota road graph, with  $K = 100$ ,  $J = 4$  scales. (a) Vertex at which wavelets are centered, (b) scaling function, (c)–(f) wavelets, scales 1–4.

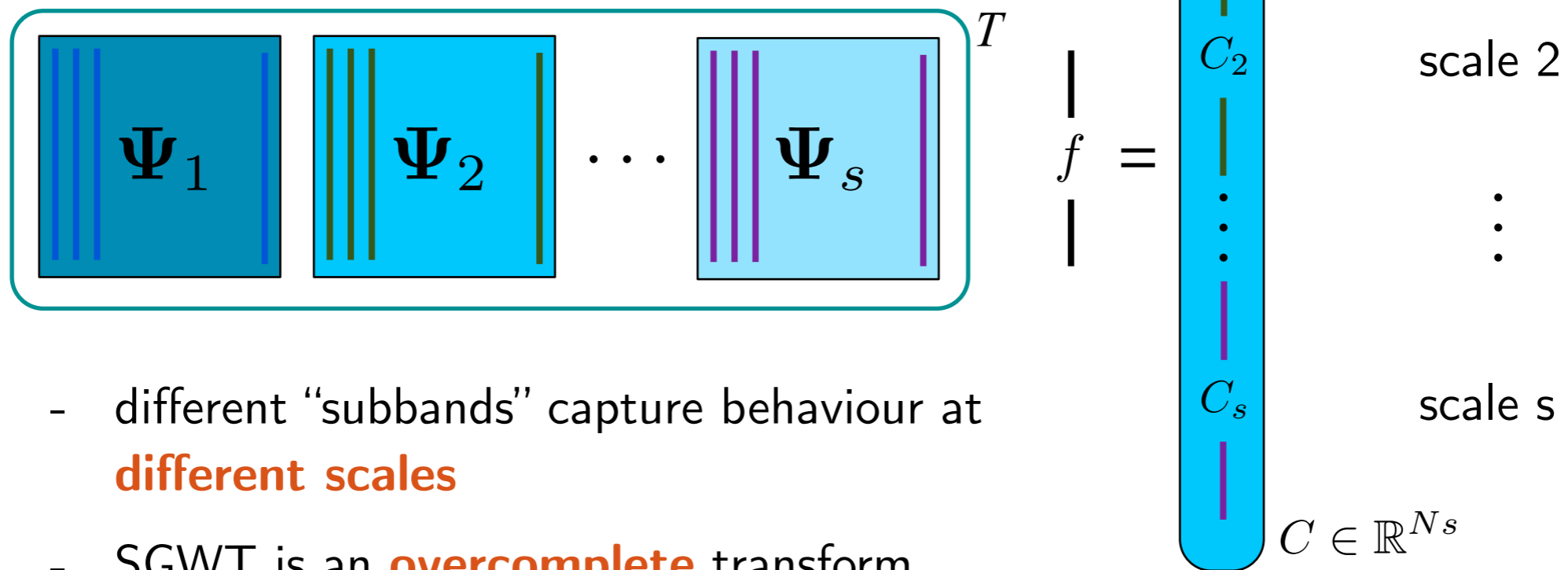
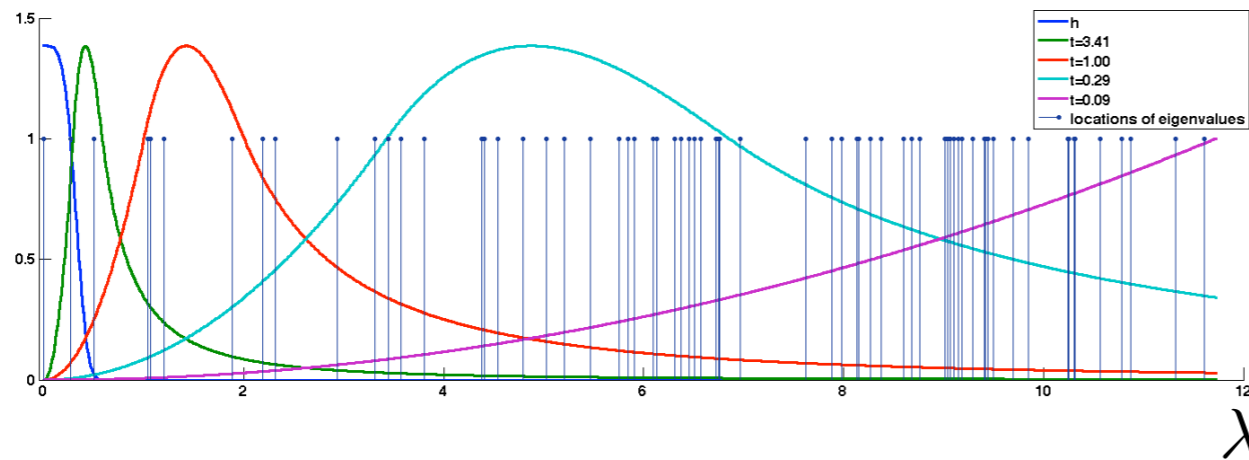


# Spectral graph wavelet transform



**Fig. 3.** Spectral graph wavelets on Swiss roll data cloud, with  $J = 4$  wavelet scales. (a) Vertex at which wavelets are centered, (b) scaling function, (c)-(f) wavelets, scales 1-4.

# Spectral graph wavelet transform



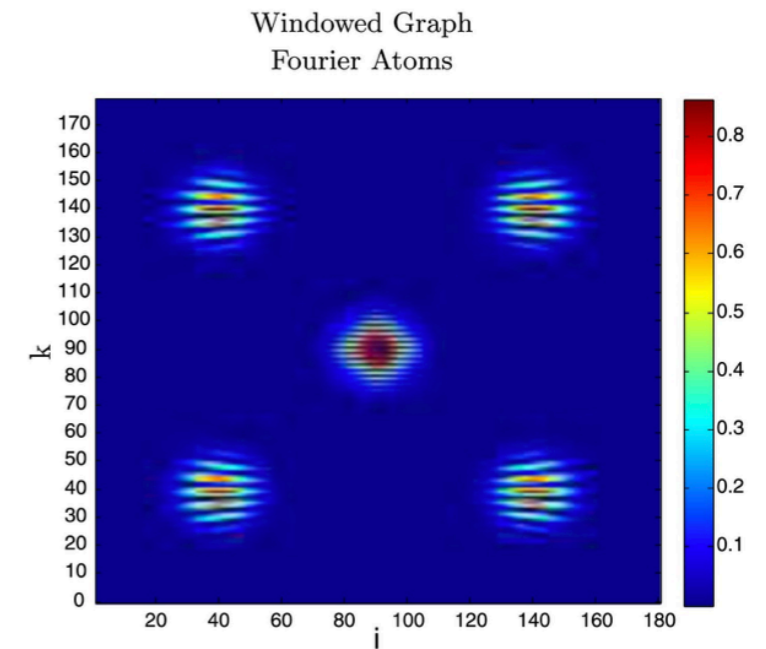
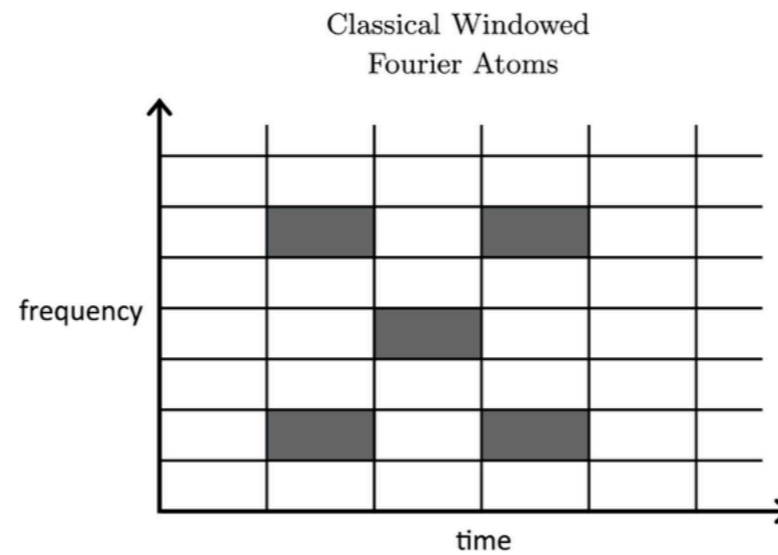
- different “subbands” capture behaviour at **different scales**
- SGWT is an **overcomplete** transform

# WGFT vs SGWT atoms

## WGFT atom

$$g_{i,k}(n) := (M_k T_i g)(n)$$

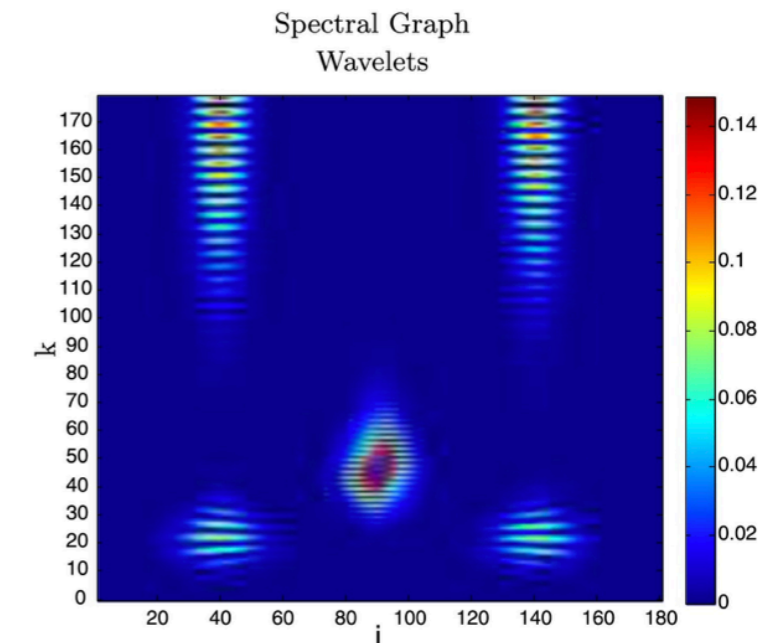
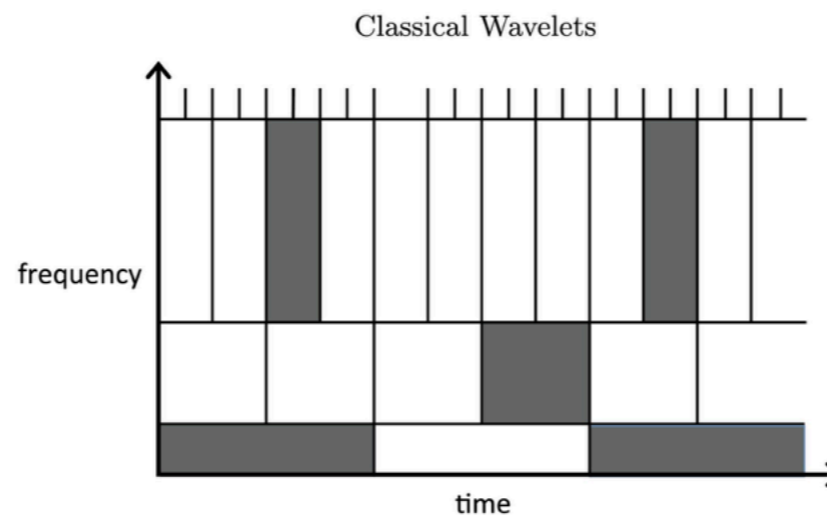
$$= N \chi_k(n) \sum_{\ell=0}^{N-1} \hat{g}(\lambda_\ell) \chi_\ell^*(i) \chi_\ell(n)$$



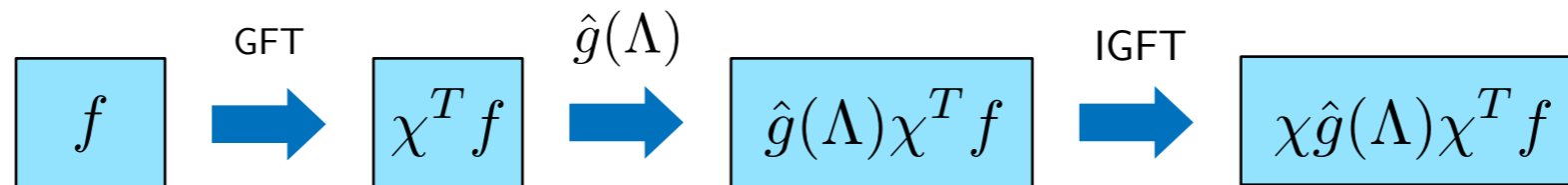
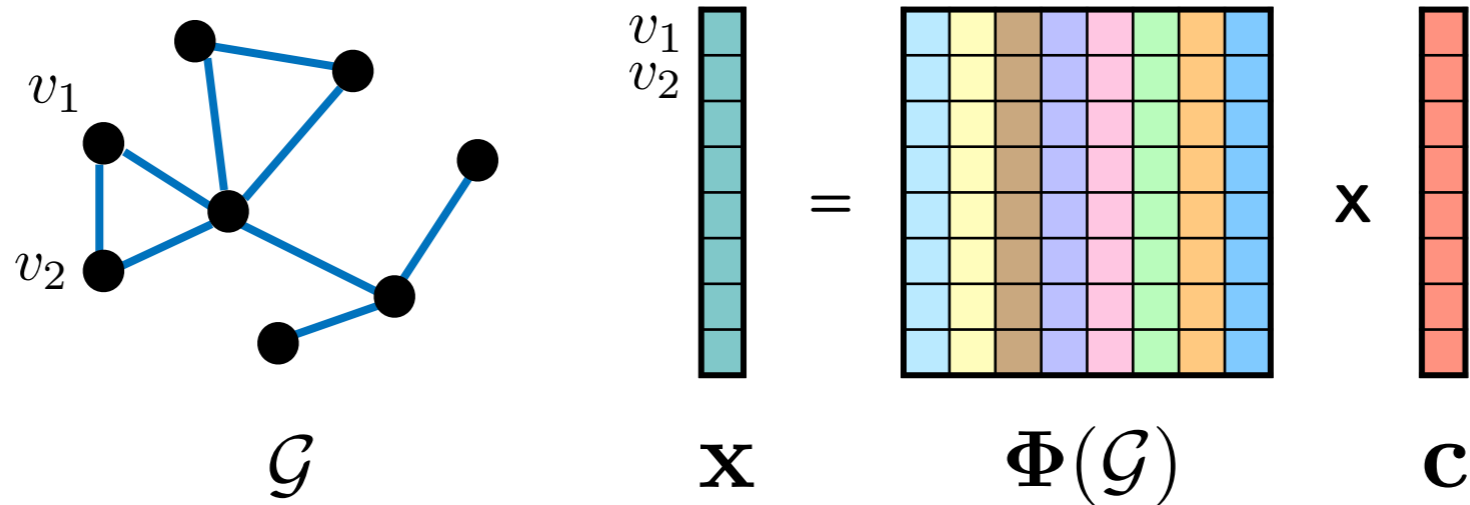
## SGWT atom

$$\psi_{i,s}(n) := (T_i D_s g)(n)$$

$$= \sum_{\ell=0}^{N-1} \hat{g}(s \lambda_\ell) \chi_\ell^*(i) \chi_\ell(n)$$



# From analytical to trained graph dictionaries



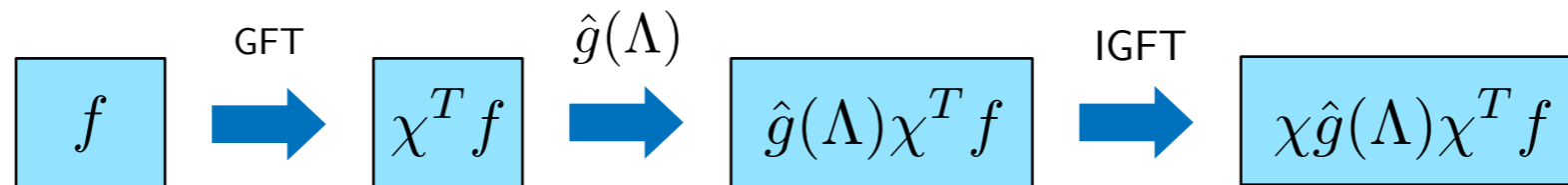
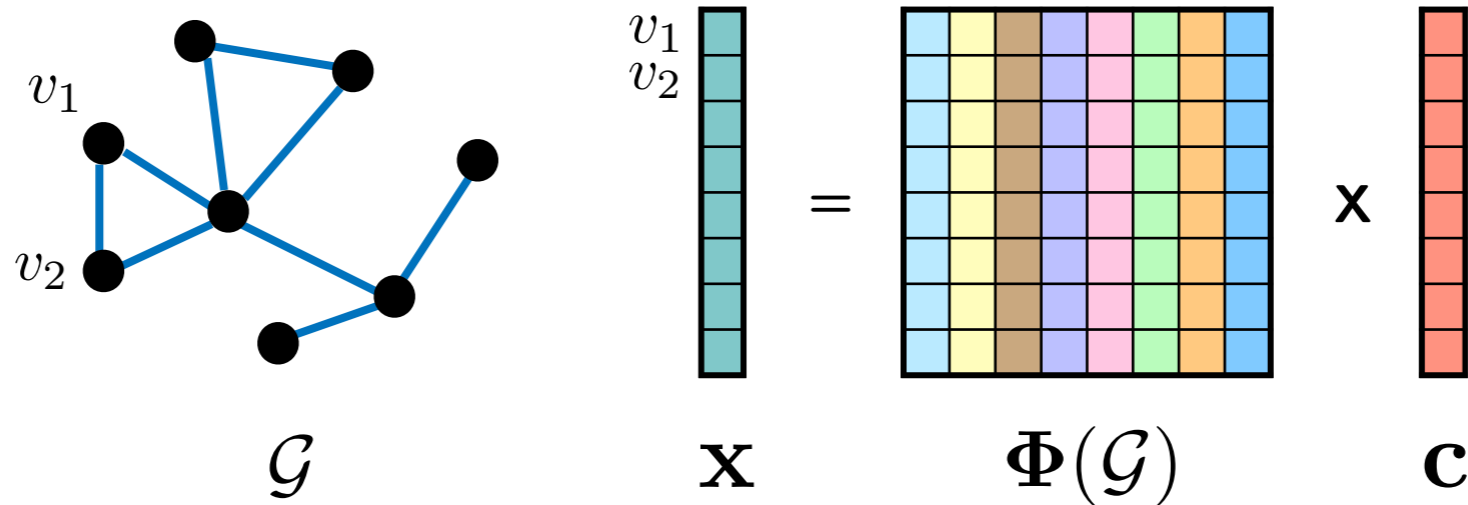
analytical graph dictionaries

$$\Phi(\mathcal{G}) = \hat{g}(L) = \chi \hat{g}(\Lambda) \chi^T$$

trained graph dictionaries

$\Phi(\mathcal{G}, \mathbf{x})$  dictionary learning on graphs?

# From analytical to trained graph dictionaries



analytical graph dictionaries

$$\Phi(\mathcal{G}) = \hat{g}(L) = \chi \hat{g}(\Lambda) \chi^T$$

trained graph dictionaries

$\Phi(\mathcal{G}, \mathbf{x}) \rightarrow$  learning  $\hat{g}(\lambda)$  by adapting to  $\mathbf{x}$

# A parametric graph dictionary

learning a parametric kernel:

$$\hat{g}_\theta(\lambda) = \sum_{j=0}^K \theta(j) \lambda^j, \quad \theta \in \mathbb{R}^{K+1}$$



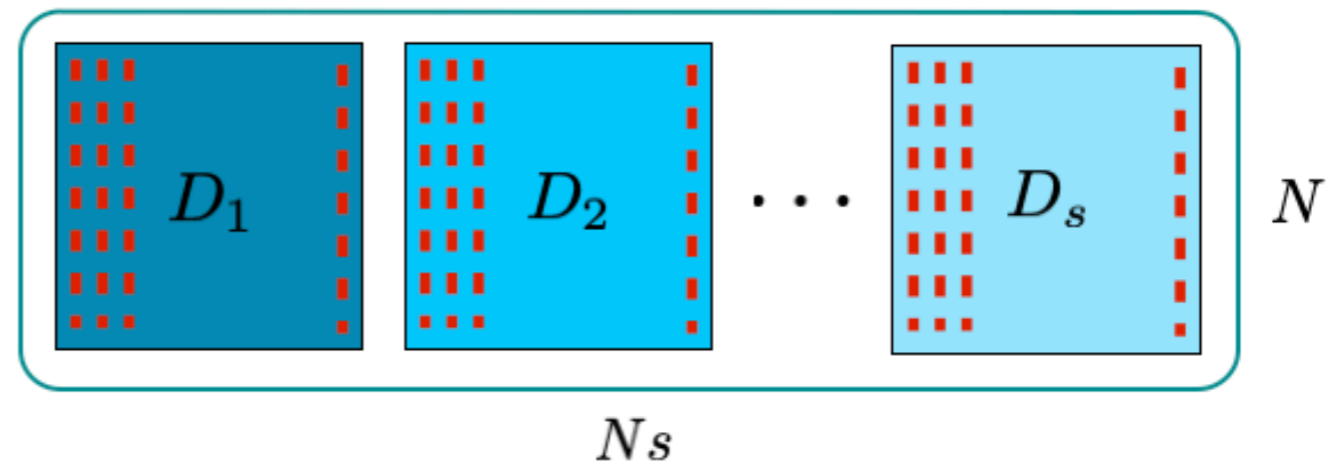
powers of graph Laplacian  
guarantee localisation

$$\hat{g}_\theta(\Lambda) = \sum_{j=0}^K \theta(j) \Lambda^j$$

consider the following dictionary:

$$D = [D_1 \ D_2 \ \cdots \ D_s] = [\chi \hat{g}_{\theta_1}(\Lambda) \chi^T \ \chi \hat{g}_{\theta_2}(\Lambda) \chi^T \ \cdots \ \chi \hat{g}_{\theta_s}(\Lambda) \chi^T]$$

- several filters to identify different **localised** spectral components



# A parametric graph dictionary

**objective:**

$$\min_{\{\theta_i\}_{i=1}^s \in \mathbb{R}^{K+1}, C \in \mathbb{R}^{NS \times M}} \|X - DC\|_F^2 + \mu \sum_{i=1}^s \|\theta_i\|_F^2$$

regularisation

adaptation to data

subject to

$$D = [\chi \hat{g}_{\theta_1}(L) \chi^T \quad \chi \hat{g}_{\theta_2}(L) \chi^T \quad \cdots \quad \chi \hat{g}_{\theta_s}(L) \chi^T]$$

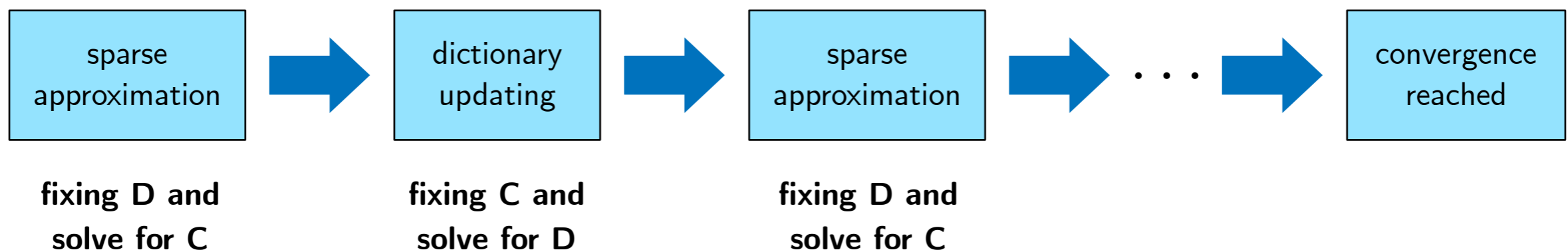
structured graph dictionaries

$$\|c_m\|_0 \leq T_0 \quad (C = [c_1 \ c_2 \ \cdots \ c_M])$$

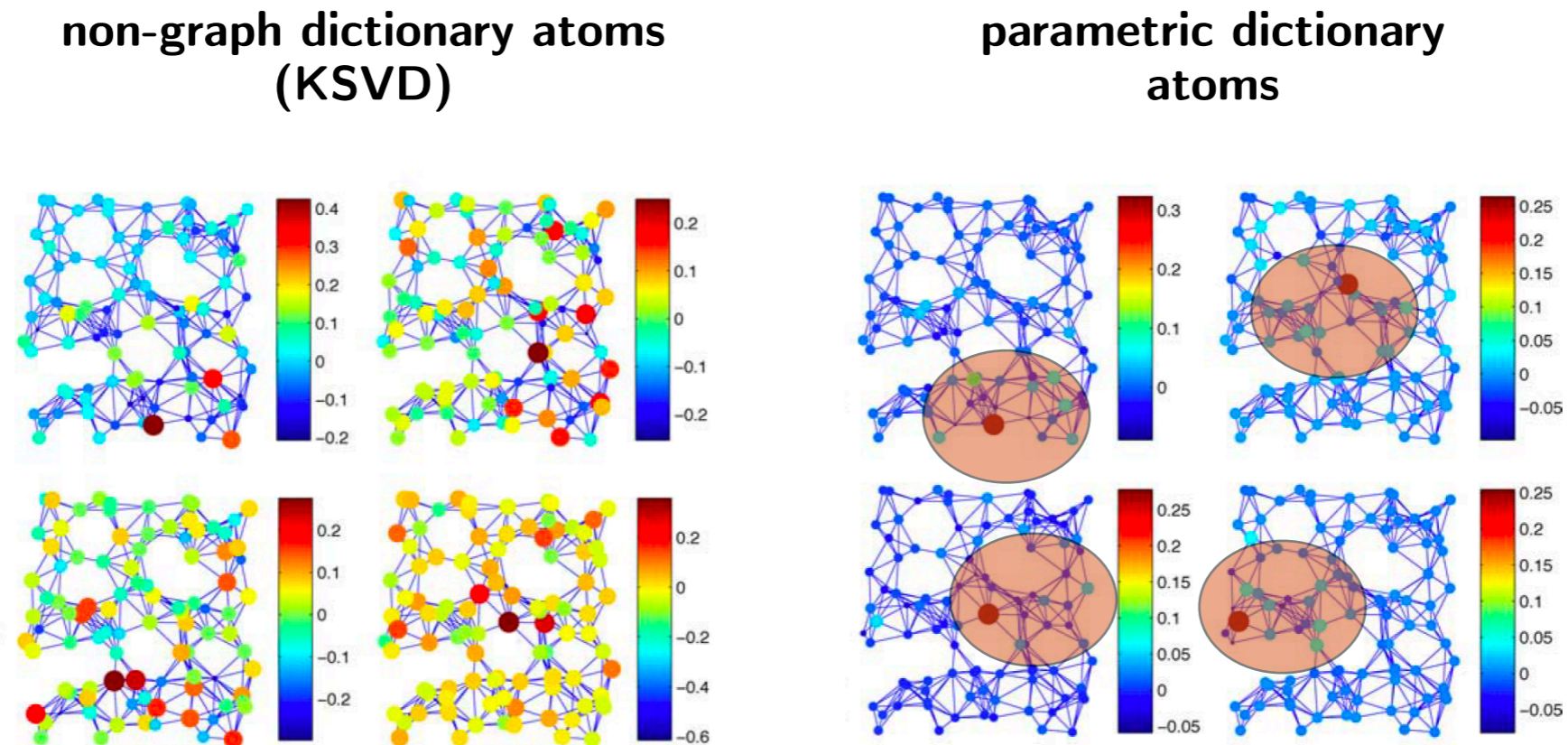
sparsity constraint

constraints guaranteeing that D is a frame

**two-stage iterative approach:**



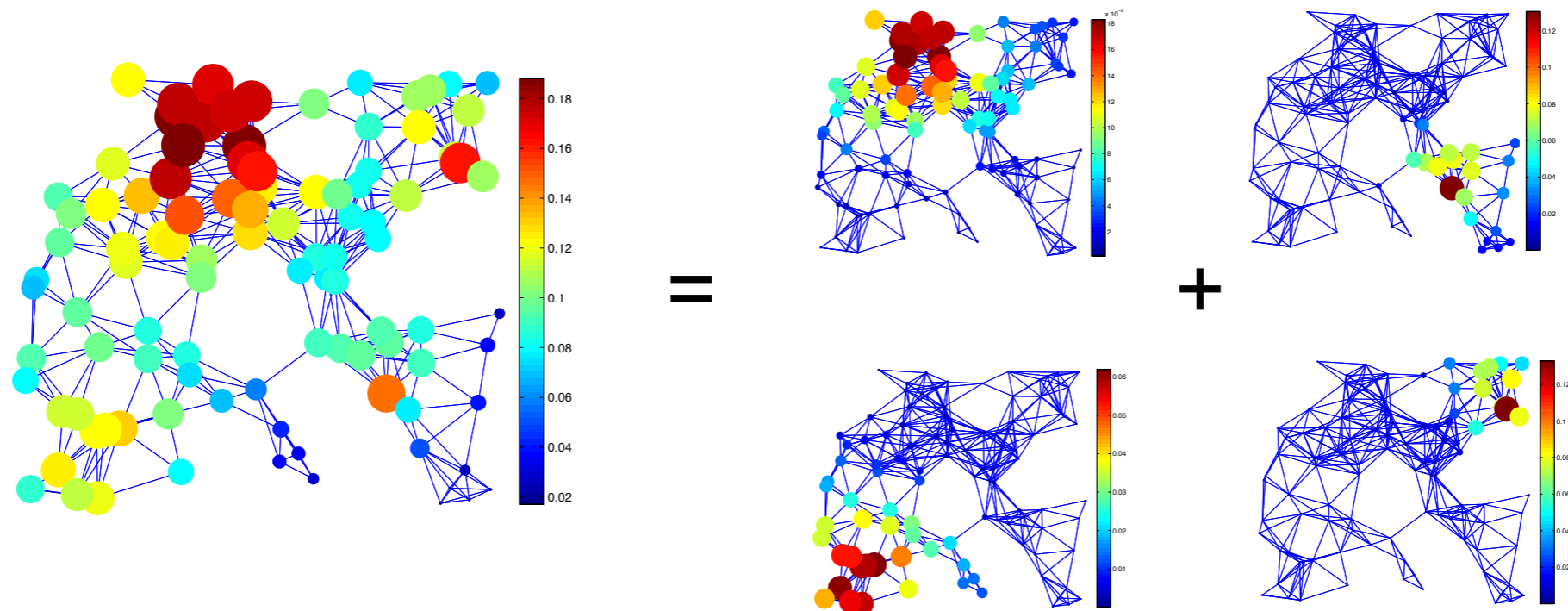
# Comparison with non-graph dictionaries



- non-graph dictionary atoms adapt to data but ignore the structure (hence are not localised)
- graph dictionary atoms adapt to data and can also be designed to be localised



# Decomposition using parametric dictionary



- the dictionary atoms adapt to localised patterns in different regions of the graph

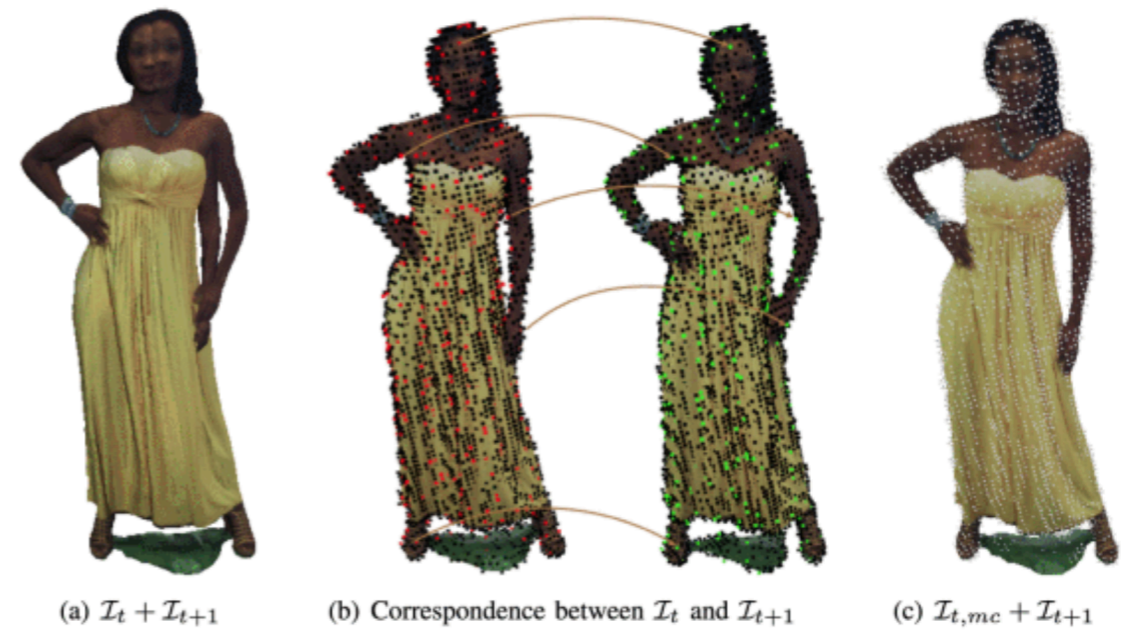
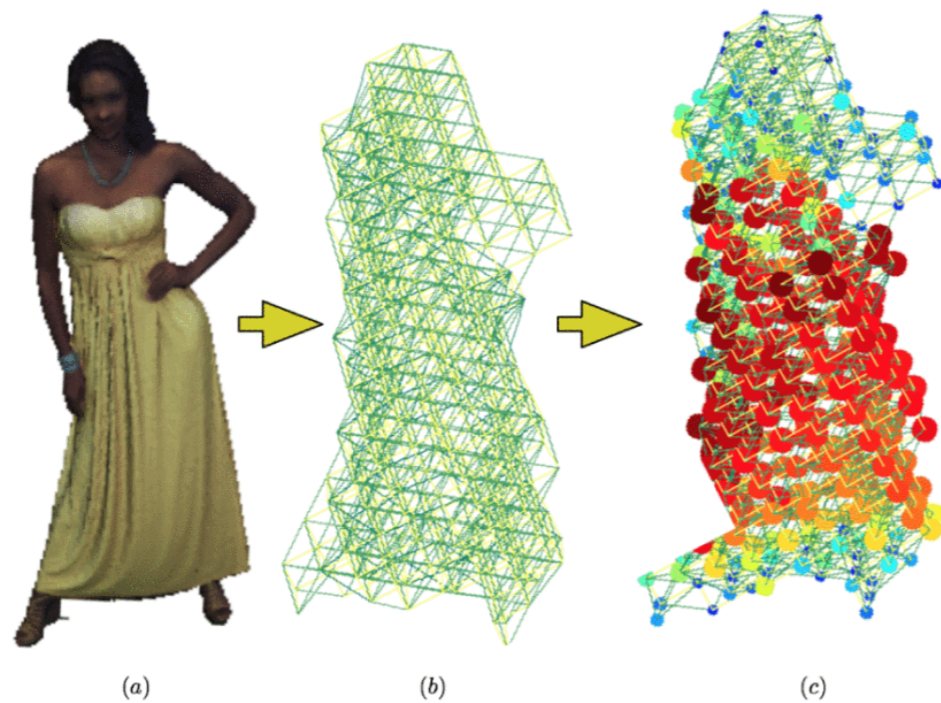
# Graph dictionary design - Summary

- Analytical vs trained graph dictionaries
  - mathematical modelling of data on graphs
  - adaptation to data on graphs
- Both approaches focus on design or learning of the kernel function
  - shift, modulation, scaling, learning-based
- This lecture has focused on Laplacian spectrum based designs
  - other possibilities exist (e.g., purely vertex-domain designs)
- Connection with other fields
  - representation learning on graphs (e.g., node embedding)
  - deep learning on graphs

# Outline

- Graph signal processing (GSP): Basic concepts
- Graph spectral filtering: Basic tools of GSP
- Connection with literature
- Representation of graph signals
- Applications

# Application I: 3D point cloud analysis

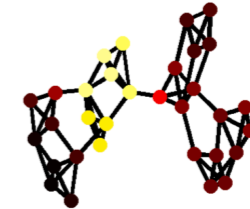
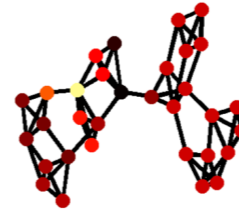


# Application II: Community detection

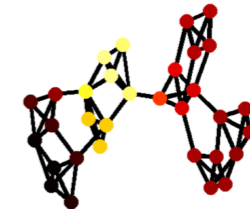
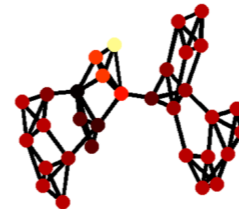
spectral graph wavelets  
at different scales:

$$D_s(a, b) = 1 - \frac{\psi_{s,a}^\top \psi_{s,b}}{\|\psi_{s,a}\|_2 \|\psi_{s,b}\|_2}.$$

NODE  
A:



NODE  
B:



CORR.  
COEF.:

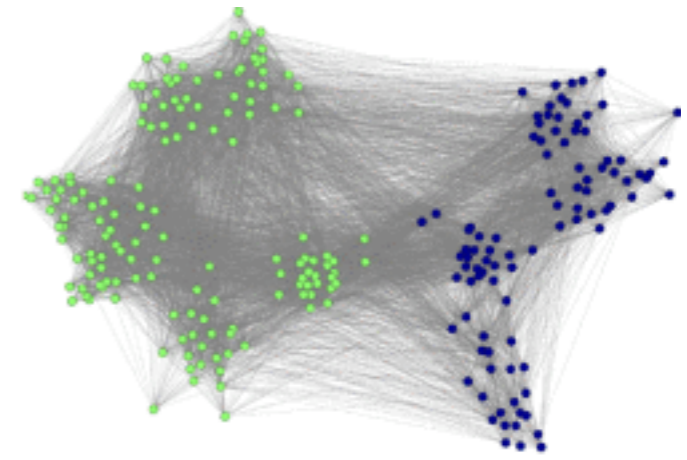
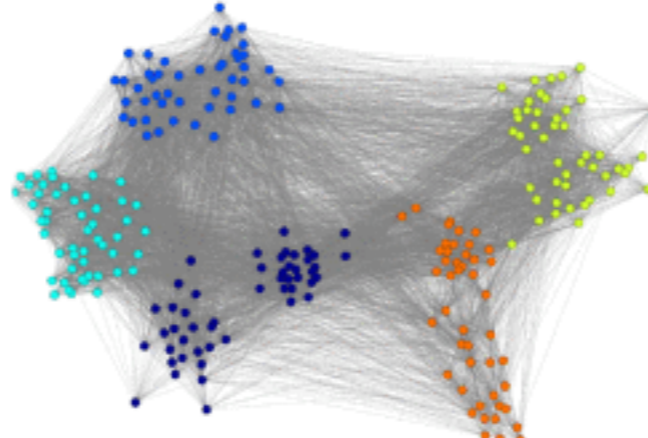
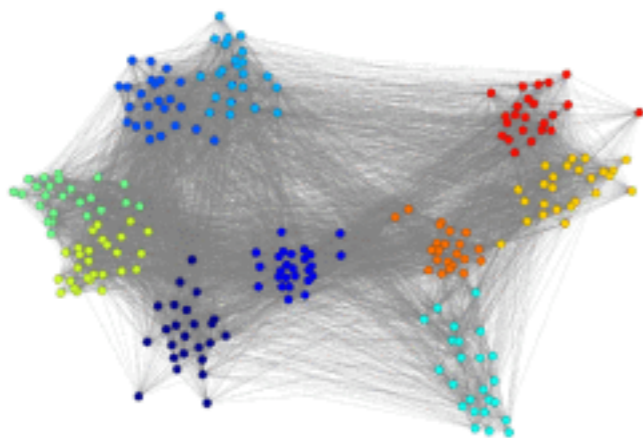
-0.50

0.97

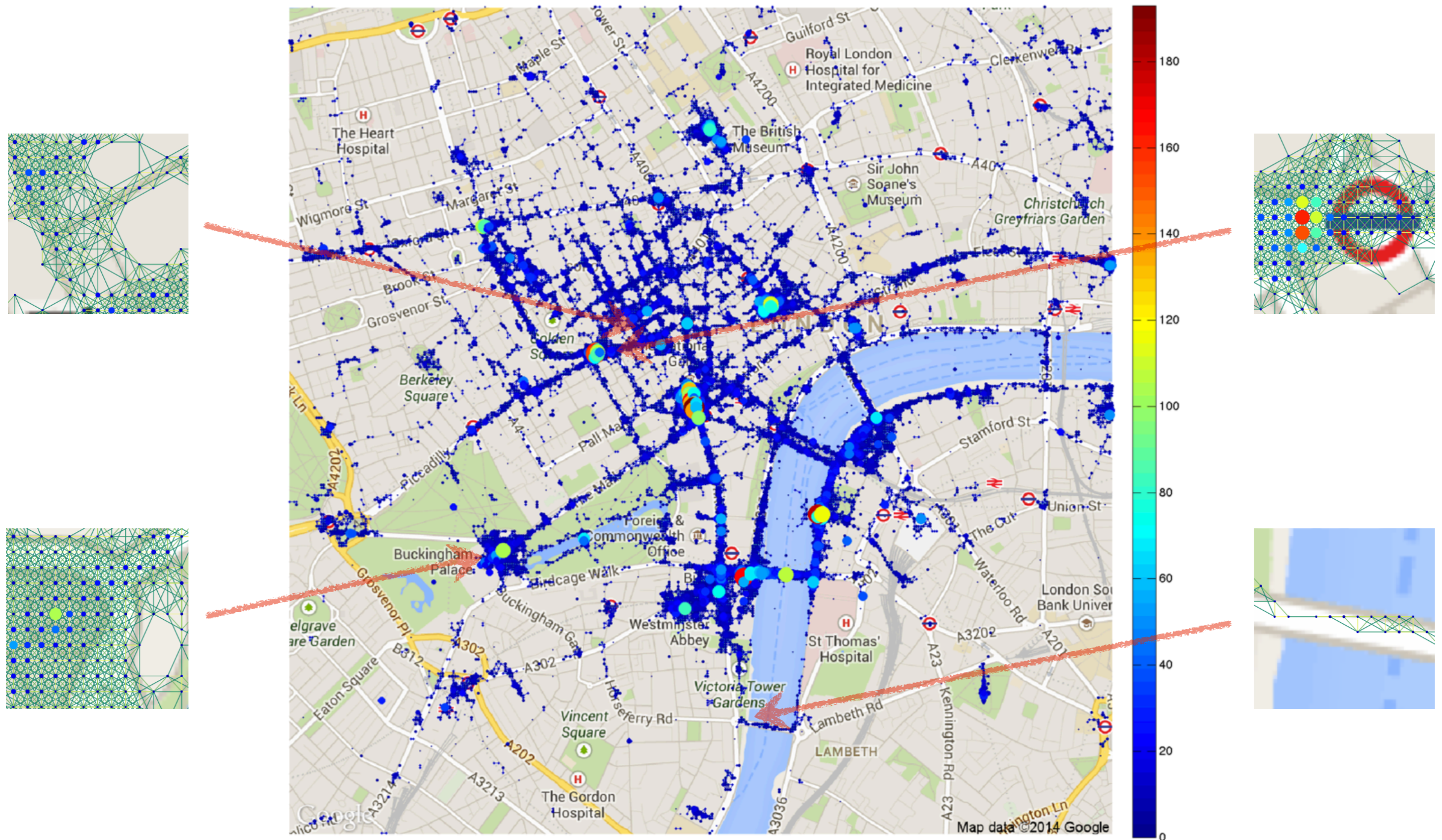
**small scale**

**large scale**

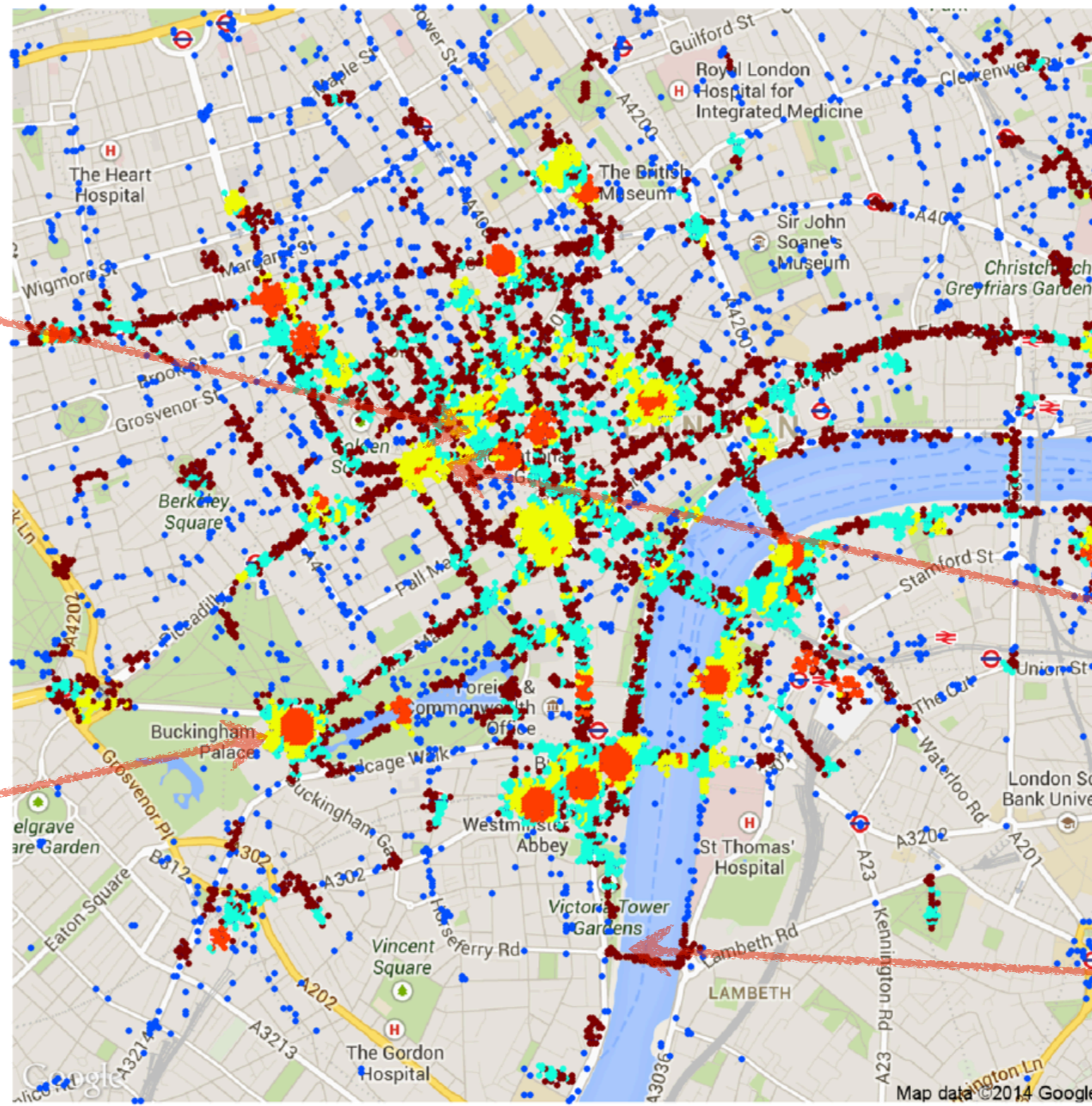
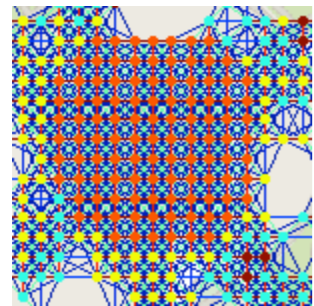
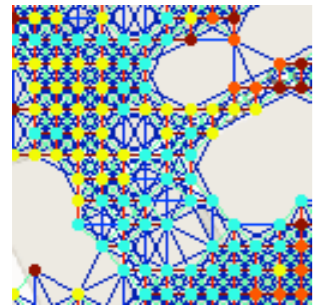
multi-scale community  
detection:



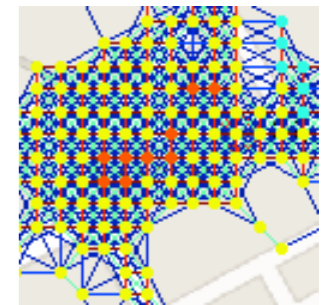
# Application III: Mobility inference



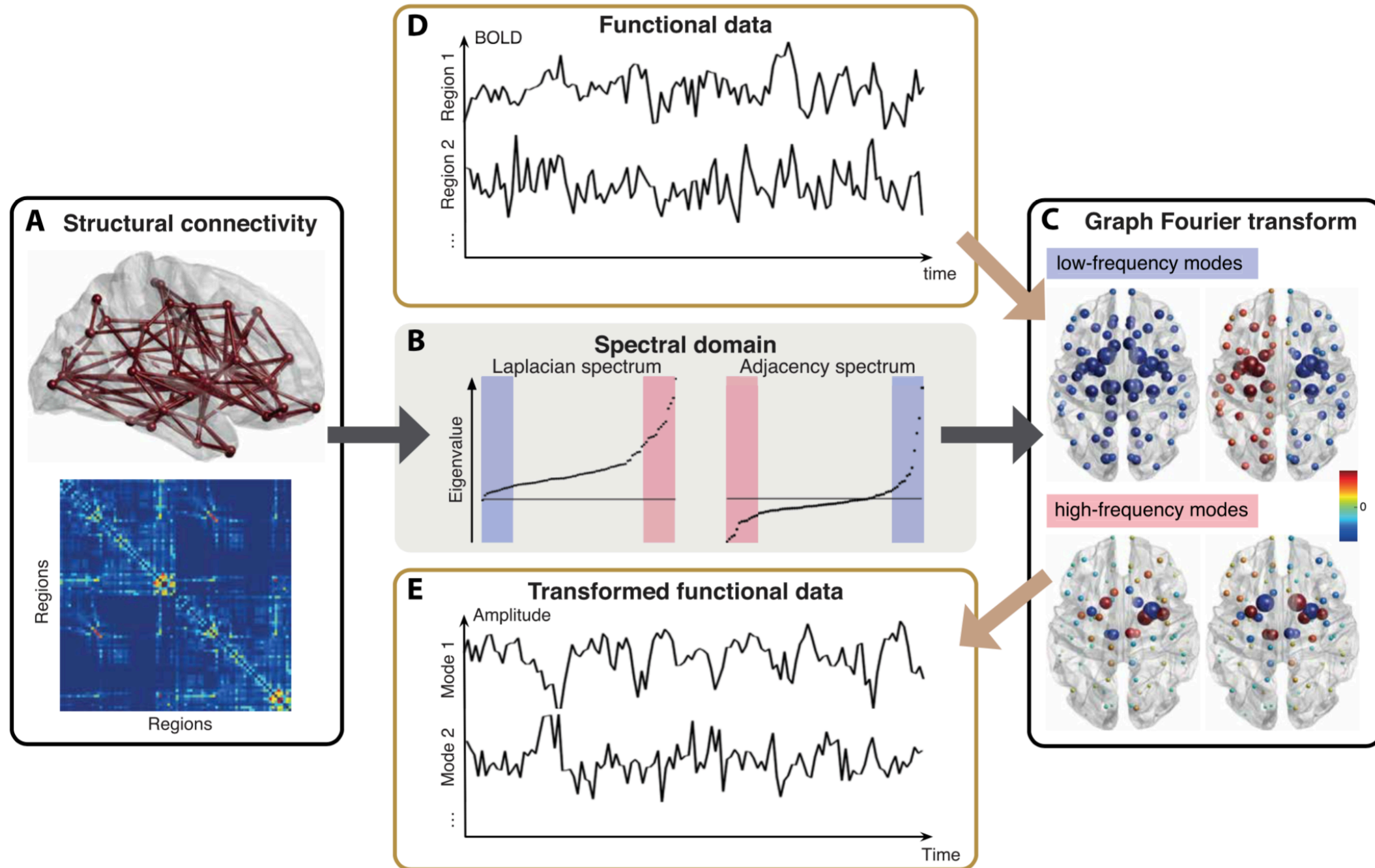
# Application III: Mobility inference



- Bi-directional
- Gathering
- Spreading
- Random



# Application IV: Neuroscience



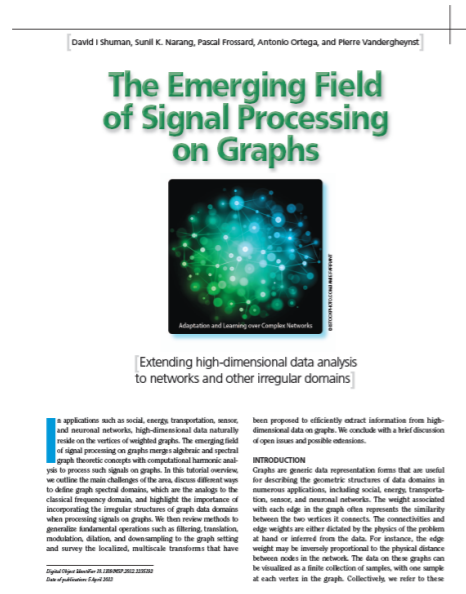


# Future of GSP

- Mathematical models for graph signals
  - global and local smoothness / regularity
  - incorporating underlying physical processes
- Graph construction
  - how to infer graph topology given observed data?
  - how to handle temporal dynamics?
- Fast implementation
  - fast graph Fourier transform
  - distributed processing
- Connection with other fields
  - machine learning on graphs
  - complex networks and systems
- Applications

# References

- Four tutorial/overview papers and one textbook



**The Emerging Field of Signal Processing on Graphs**  
David I. Shuman, Sunil K. Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst

Extending high-dimensional data analysis to networks and other irregular domains

In applications such as social, energy transportation, sensor, and neuronal networks, high-dimensional data naturally reside on the vertices of weighted graphs. The emerging field of signal processing on graphs merges algebraic and spectral graph theoretic concepts with computational harmonic analysis to process such signals on graphs. In this tutorial overview, we outline the main challenges of the area, discuss different ways to define graph spectral domains, which are the analogs to the classical frequency domain, and highlight the importance of incorporating the irregular structure of graph data domains when processing signals on graphs. We then review methods to generate fundamental operators such as filtering, translation, modulation, dilation, and down-sampling to the graph setting and survey the localized, multiscale transforms that have been proposed to effectively extract information from high-dimensional data on graphs. We conclude with a brief discussion of open issues and possible extensions.

**INTRODUCTION**  
Graphs are generic data representation forms that are useful for describing the geometric structure of data domains in numerous applications, including social, energy, transportation, sensor, and neuronal networks. The weight associated with each edge in the graph often represents the similarity between the two vertices it connects. The connectivities and edge weights are either dictated by the physics of the problem at hand or inferred from the data. For instance, the edge weight may be inversely proportional to the physical distance between nodes in the network. The data on these graphs can be visualized as a finite collection of samples, with one sample at each vertex in the graph. Collectively, we refer to the



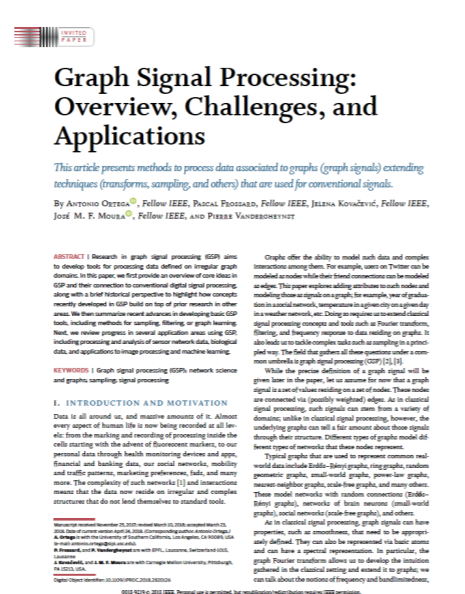
**Vertex-frequency analysis on graphs**  
David I. Shuman<sup>\*,†</sup>, Benjamin Recht<sup>‡</sup>, Pierre Vandergheynst<sup>§</sup>

<sup>\*</sup> Department of Mathematics, Statistics, and Computer Science, Massachusetts College of Saint Paul, MA, USA  
<sup>†</sup> Signal Processing Laboratory (LTSI), Ecole Polytechnique Palaiseau de Lausanne (EPFL), Lausanne, Switzerland

**ABSTRACT**  
One of the key challenges in the area of signal processing on graphs is to design dictionaries and transform methods to identify and exploit structure in signals on weighted graphs. To do so, we need to account for the intrinsic geometric structure of the underlying graph data domains. In this paper, we present one of the most important signal processing tools – windowed Fourier analysis – in the graph setting. Our approach is to first define generalized translation, modulation, and modulation operators for signals on graphs, and explore related properties such as the localization of translated and modulated graph bands. We then use these operators to define a windowed graph Fourier transform, making vertex-frequency analysis. When we apply this transform to a signal with frequency component that vary along a path graph, the resulting spectrogram matches our intuition from classical discrete-time signal processing. Yet, our construction is fully generalized and can be applied to analyze signals on any undirected, connected, weighted graph.

**1. Introduction**  
In applications such as social networks, electricity networks, transportation networks, and sensor networks, data naturally reside on the vertices of weighted graphs. Moreover, weighted graphs are a flexible tool that can be used to describe similarities between data points in structured learning problems, functional connectivities between different regions of the brain, and the geometric structure of complex other topologically-complex data domains.

<sup>\*</sup> This work was supported by FET-Open grant number 255031 UNLxO.  
<sup>†</sup> Part of the work reported here was presented at the IEEE Statistical Signal Processing Workshop, August 2012, Ann Arbor, MI.  
<sup>‡</sup> Corresponding author.  
<sup>§</sup> E-mail addresses: shuman@mathcs.dartmouth.edu (D.I. Shuman), benjamin.recht@epfl.ch (B. Recht), pierre.vandergheynst@epfl.ch (P. Vandergheynst).  
<sup>¶</sup> The authors would like to thank the anonymous reviewers for their constructive comments on this article.  
<http://dx.doi.org/10.1109/ICASSP.2015.452.02.005>  
1063-3200/15/2015 IEEE. All rights reserved.



## Graph Signal Processing: Overview, Challenges, and Applications

This article presents methods to process data associated to graphs (graph signals) extending techniques (transforms, sampling, and others) that are used for conventional signals.

By ANTONIO ORTEGA<sup>\*,†</sup>, Fellow IEEE, PASCAL FROSSARD, Fellow IEEE, JITENDRA KOVAČEVIĆ, Fellow IEEE, JOSÉ M. F. MOURA<sup>\*,‡</sup>, Fellow IEEE, and PIERRE VANDERGHEYNST

**ABSTRACT** | Research in graph signal processing (GSP) aims to develop tools for processing data defined on irregular graph domains. In this paper, we first provide an overview of core ideas in GSP and their connection to conventional digital signal processing, along with a brief historical perspective to highlight how concepts recently developed in GSP build on top of prior research in other areas. We then summarize recent advances in developing basic GSP tools, including methods for sampling, filtering, or graph learning. Next, we review progress in several application areas using GSP, including processing and analysis of sensor network data, biological data, and applications to image processing and machine learning.

**KEYWORDS** | Graph signal processing (GSP), network science and graphs, sampling, signal processing

**1. INTRODUCTION AND MOTIVATION**  
Data is all around us, and massive amounts of it. Almost every aspect of human life is now being recorded at all levels: from the mailing and recording of processing inside the cells starting with the advent of bio-sensing analysis, to our personal data through health monitoring devices and apps, financial and banking data, our social networks, mobility and traffic patterns, marketing preferences, fads, and many more. The complexity of such networks [1] and interactions means that the data now reside on irregular and complex structures that do not lend themselves to standard tools.

**INTRODUCTION**  
Data is all around us, and massive amounts of it. Almost every aspect of human life is now being recorded at all levels: from the mailing and recording of processing inside the cells starting with the advent of bio-sensing analysis, to our personal data through health monitoring devices and apps, financial and banking data, our social networks, mobility and traffic patterns, marketing preferences, fads, and many more. The complexity of such networks [1] and interactions means that the data now reside on irregular and complex structures that do not lend themselves to standard tools.

808 PROCEEDINGS OF THE IEEE | Vol. 103, No. 5, May 2015

Xiaohua Dong, Dorcas Thano, Laura Toli, Michael Bronstein, and Pascal Frossard



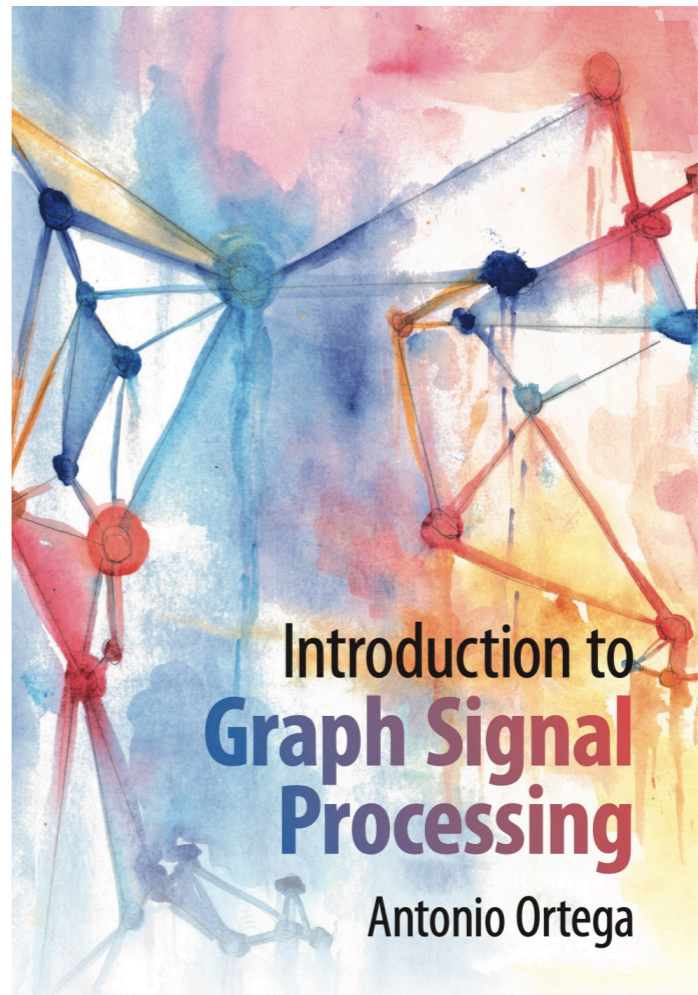
## Graph Signal Processing for Machine Learning

A review and new perspectives

The effective representation, processing, analysis, and visualization of large-scale structural data, especially those related to complex domains, such as networks and graphs, are one of the key questions in modern machine learning. Graph signal processing (GSP), a vibrant branch of signal processing methods and algorithms that aims at handling data supported on graphs, opens new paths of research to address this challenge. In this article, we review a few important contributions made by GSP concepts and tools, such as graph filters and transforms, to the development of novel machine learning algorithms. In particular, our discussion focuses on the following three aspects: exploiting data structure and relational priors, improving data and computational efficiency, and enhancing model interpretability. Furthermore, we provide new perspectives on the future development of GSP techniques that may serve as a bridge between applied mathematics and signal processing on one side and machine learning and network science on the other. Cross-fertilization across these different disciplines may help tackle the numerous challenges of complex data analysis in the modern age.

**Introduction**  
We live in a connected society. Data collected from large-scale interactive systems, such as biological, social, and financial networks, become largely available. In parallel, the past few decades have seen a significant amount of interest in the machine learning community for network data processing and analysis. Networks have an intrinsic structure that conveys very specific properties to data, e.g., interdependencies between data entities in the form of pairwise relationships. These properties are traditionally captured by mathematical representations such as graphs.

In this context, new trends and challenges have been developing fast. Let us consider, for example, a network of protein-protein interactions and the expression level of individual genes at every point in time. Some typical tasks in network biology related to this type of data are: discovery of key genes (via protein grouping) affected by the infection and Z<sub>2</sub> prediction of how the host organism reacts (in terms of gene expression).



## Introduction to Graph Signal Processing

Antonio Ortega

# Resources

- Graph signal processing
  - MATLAB toolbox
    - <https://github.com/epfl-lts2/gspbox>
    - <https://github.com/STAC-USC/GraSP>
  - Python toolbox
    - <https://github.com/epfl-lts2/pygsp>
- More at: <http://www.robots.ox.ac.uk/~xdong/resource.html>