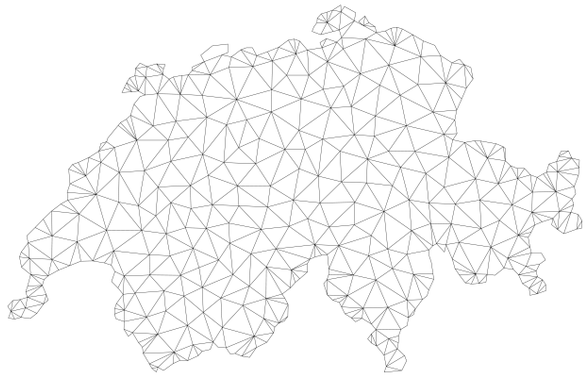
Applied Machine Learning

Graph machine learning - Part I

Xiaowen Dong
Department of Engineering Science

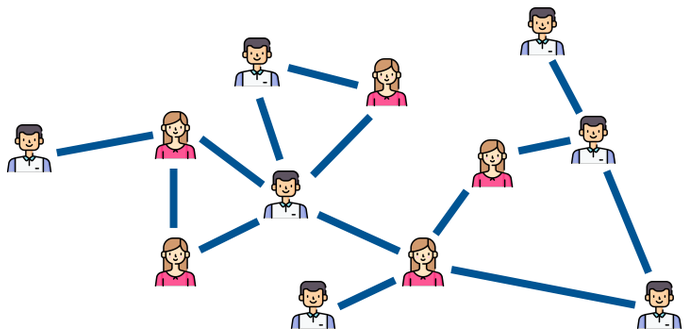
Overview

Networks are pervasive

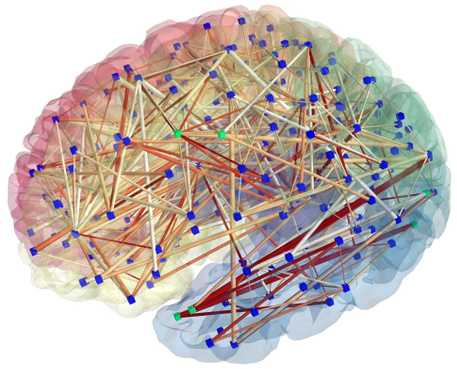


geographical network

traffic network



social network

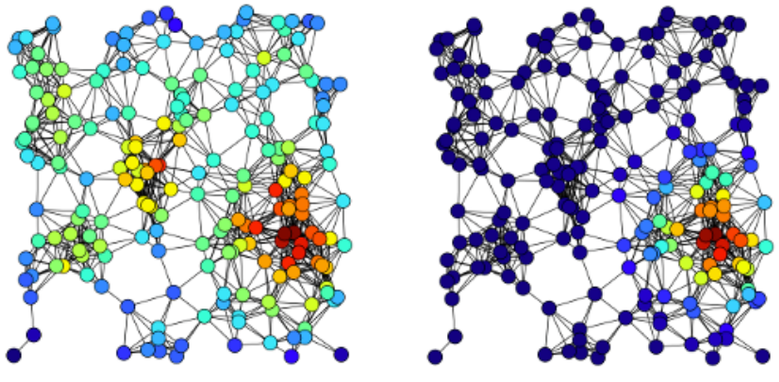


brain network

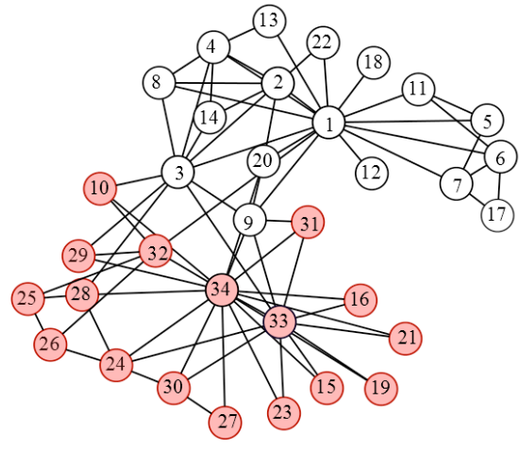
graphs provide mathematical representation of networks

The field of network science

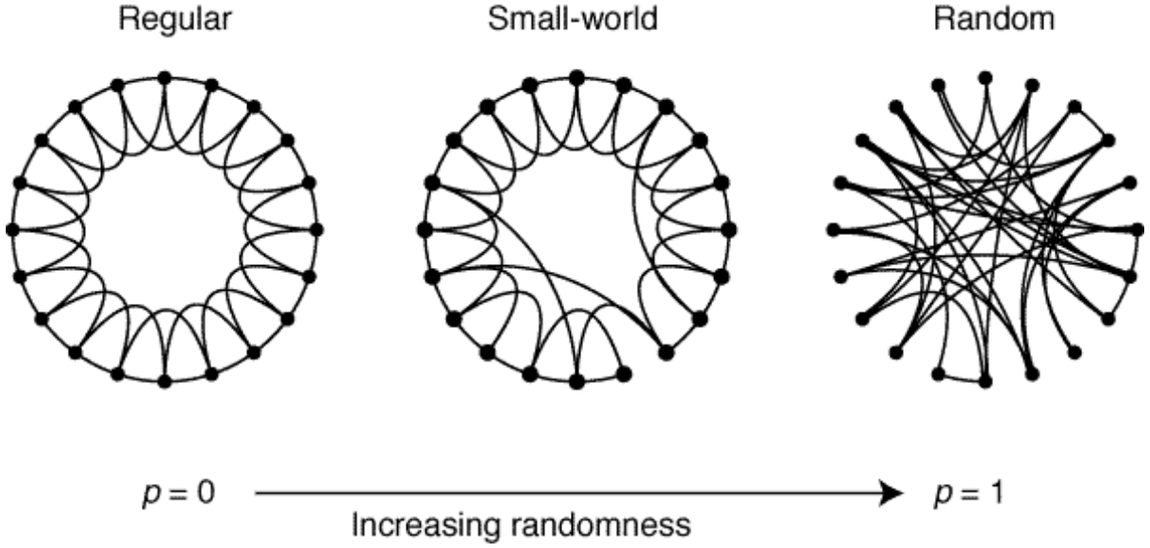
network centrality



community detection



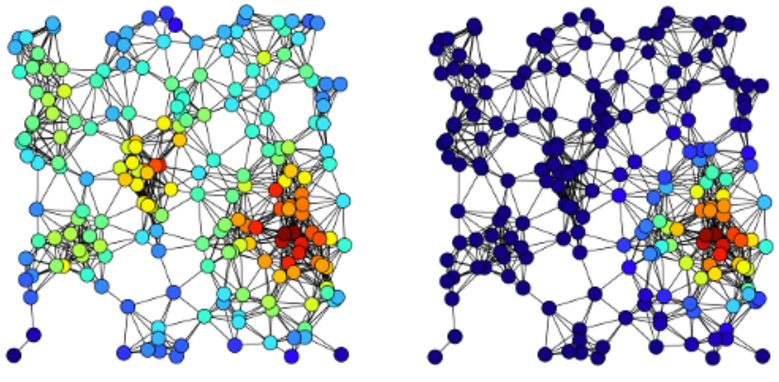
random graph models



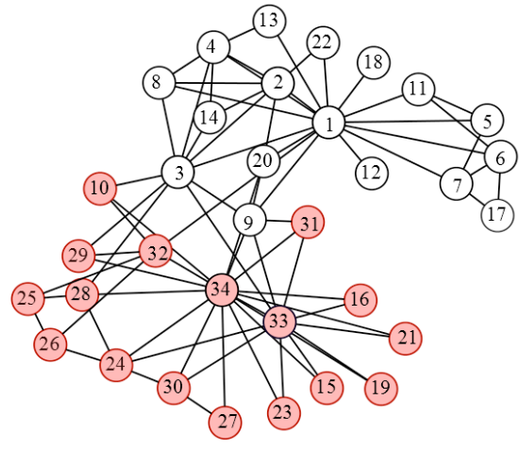
Watts and Strogatz, "Collective dynamics of 'small-world' networks," Nature, 1998.
Newman, "Networks: An introduction," Oxford University Press, 2010.

The field of network science

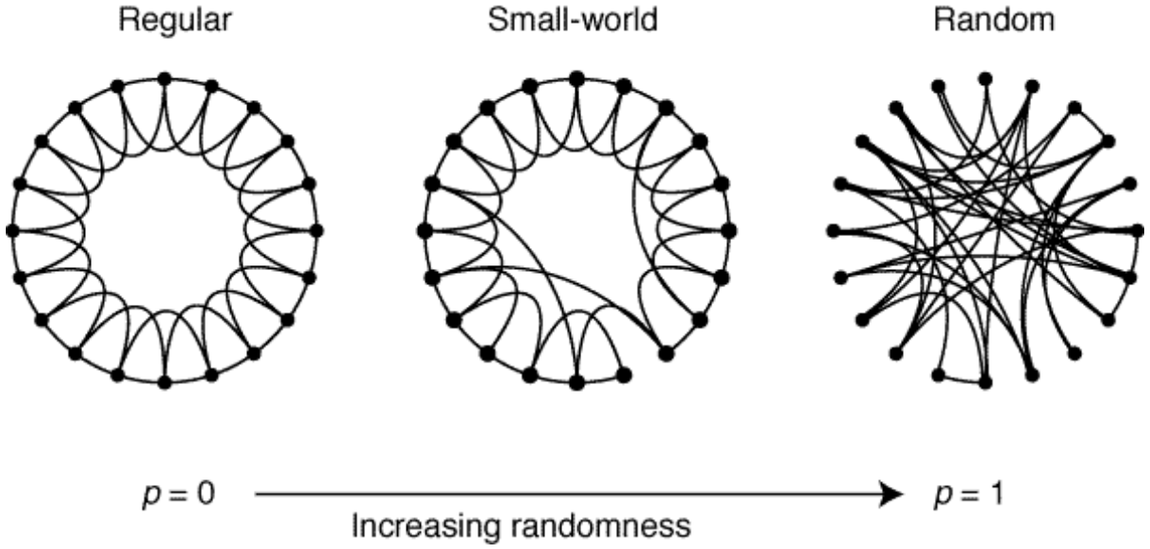
network centrality



community detection



random graph models

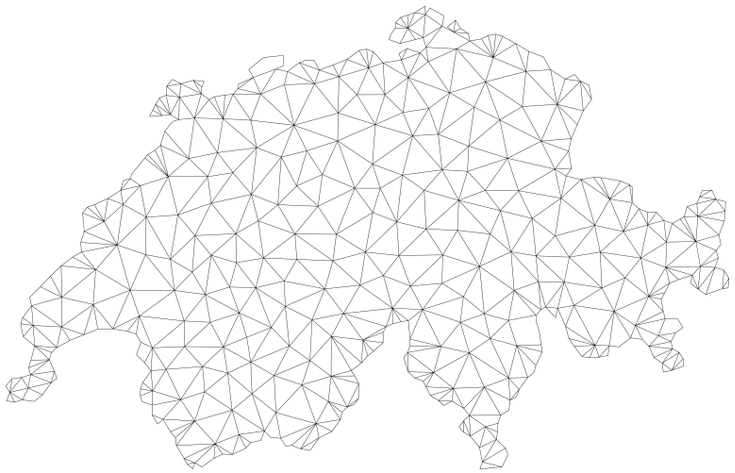


from edge attributes to node attributes

from graphs to graph-structured data

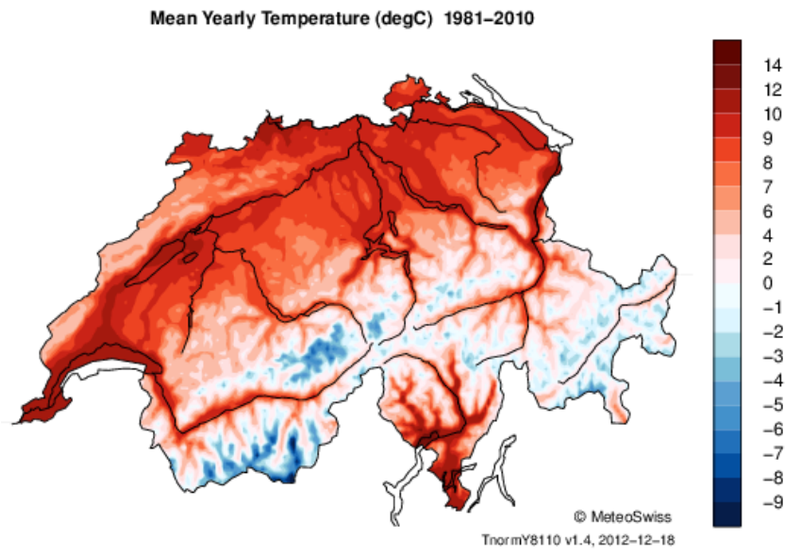
Watts and Strogatz, "Collective dynamics of 'small-world' networks," Nature, 1998.
Newman, "Networks: An introduction," Oxford University Press, 2010.

Graph-structured data are pervasive



- nodes
  - geographical regions
- edges
  - geographical proximity between regions

Graph-structured data are pervasive

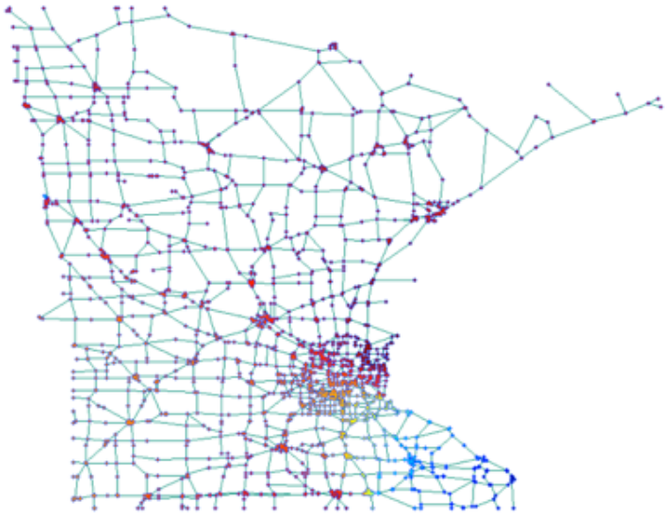


- nodes
  - geographical regions
- edges
  - geographical proximity between regions
- signal
  - temperature records in these regions

Graph-structured data are pervasive

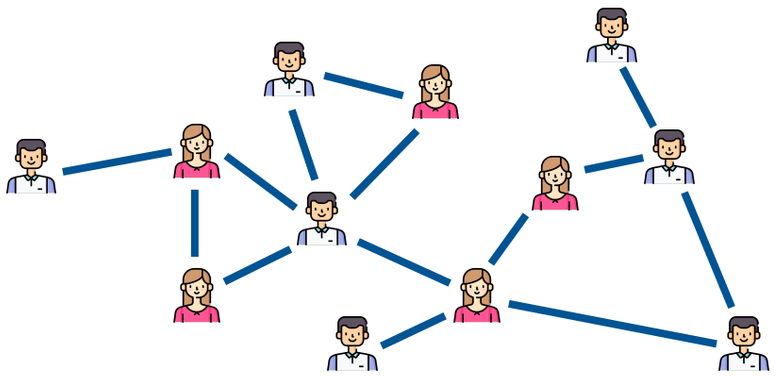
- nodes
  - road junctions
- edges
  - road connections

Graph-structured data are pervasive



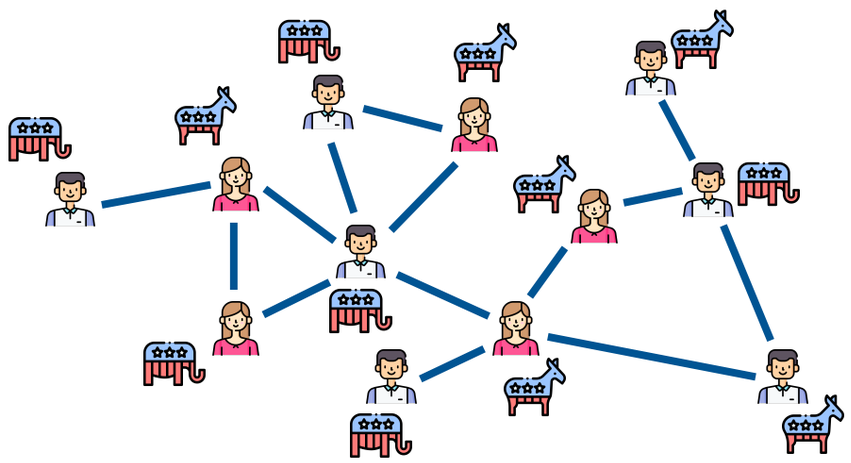
- nodes
  - road junctions
- edges
  - road connections
- signal
  - traffic congestion at junctions

Graph-structured data are pervasive



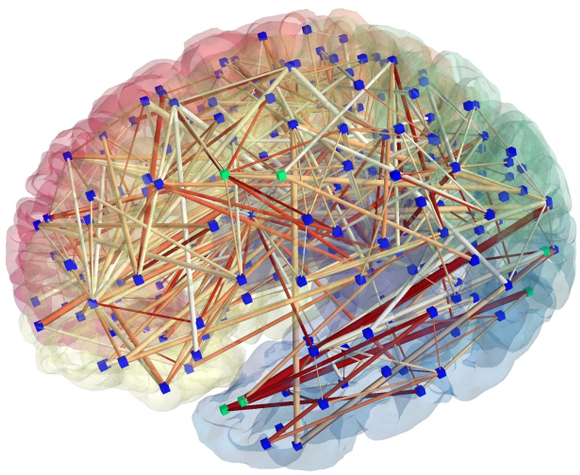
- nodes
  - individuals
- edges
  - friendship between individuals

Graph-structured data are pervasive



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

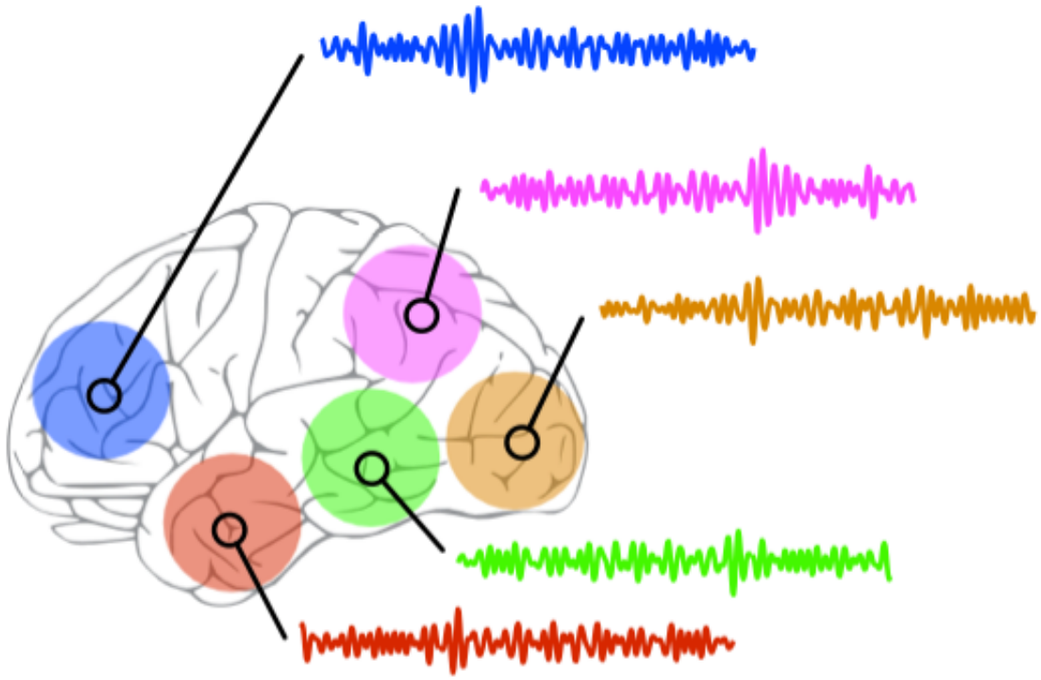
Graph-structured data are pervasive



- nodes
  - brain regions
- edges
  - structural connectivity between brain regions

Richiardi et al., "Machine learning with brain graphs," IEEE SPM, 2013.

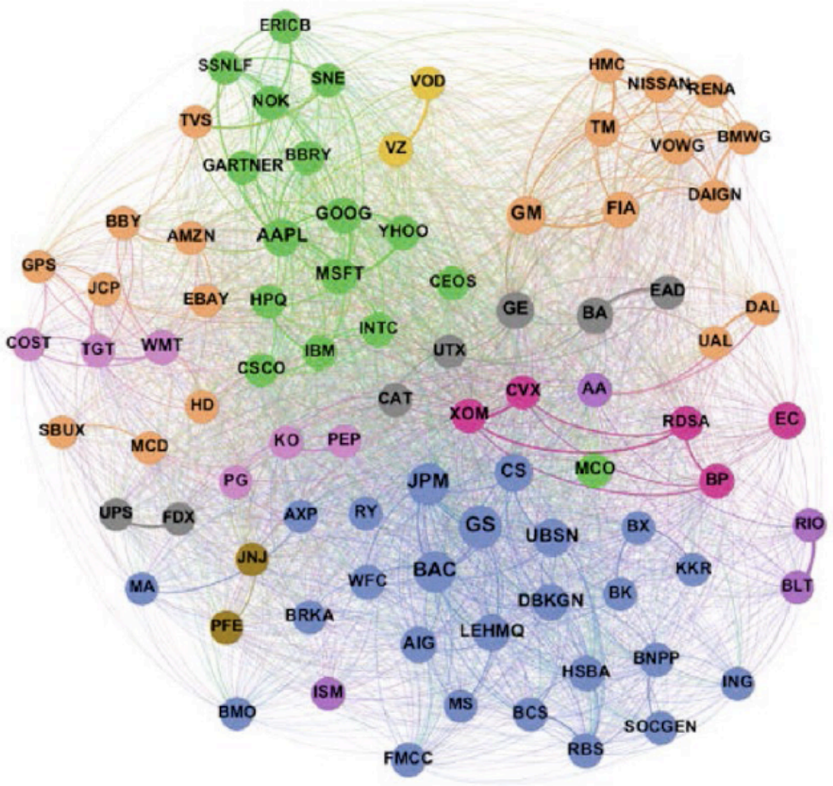
Graph-structured data are pervasive



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

Richiardi et al., "Machine learning with brain graphs," IEEE SPM, 2013.

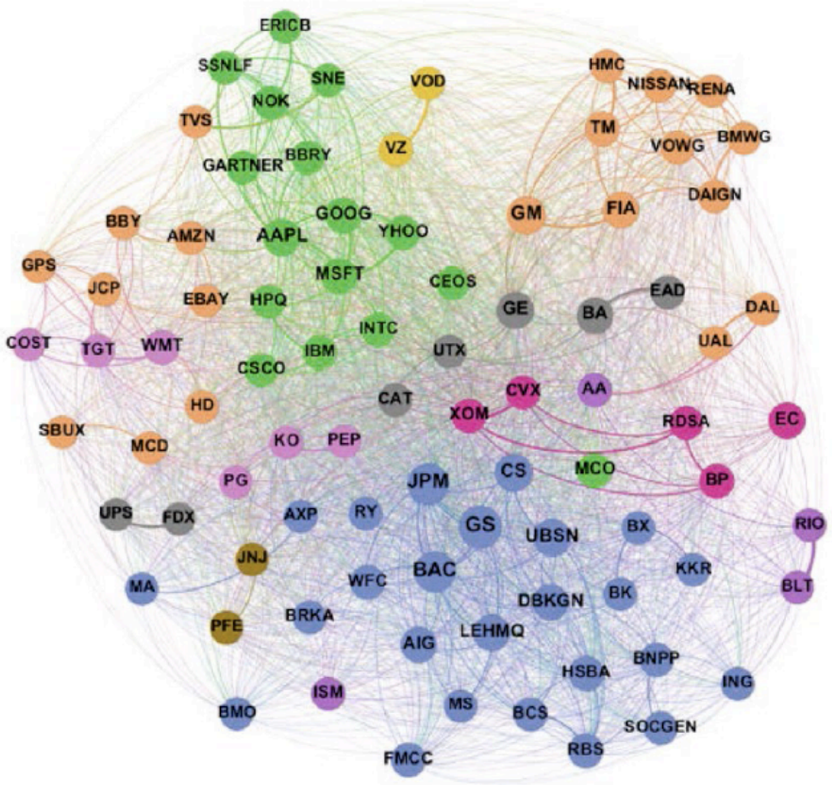
Graph-structured data are pervasive



- nodes
  - companies
- edges
  - co-occurrence of companies in financial news

Wan et al., "Sentiment correlation in financial news networks and associated market movements," Scientific Reports, 2021.

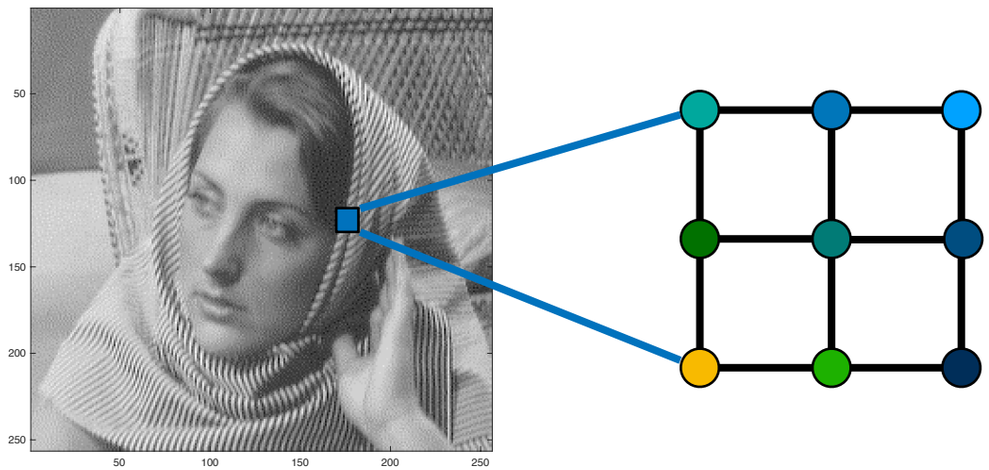
Graph-structured data are pervasive



- nodes
  - companies
- edges
  - co-occurrence of companies in financial news
- signal
  - stock prices of these companies

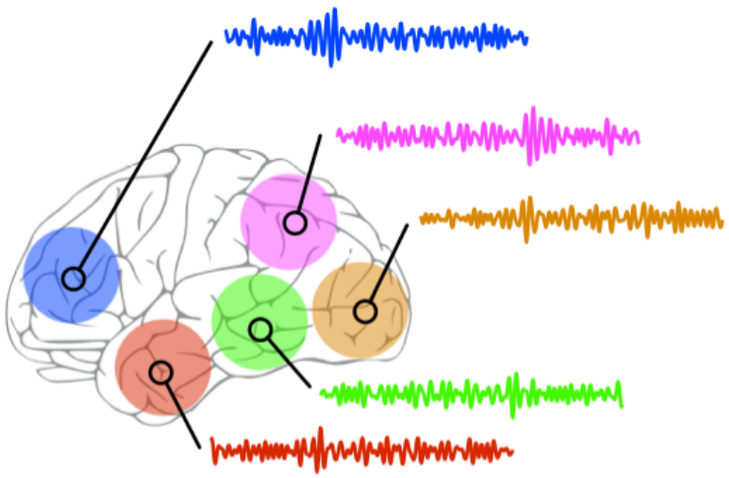
Wan et al., "Sentiment correlation in financial news networks and associated market movements," Scientific Reports, 2021.

Graph-structured data are pervasive



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

Learning with graph-structured data

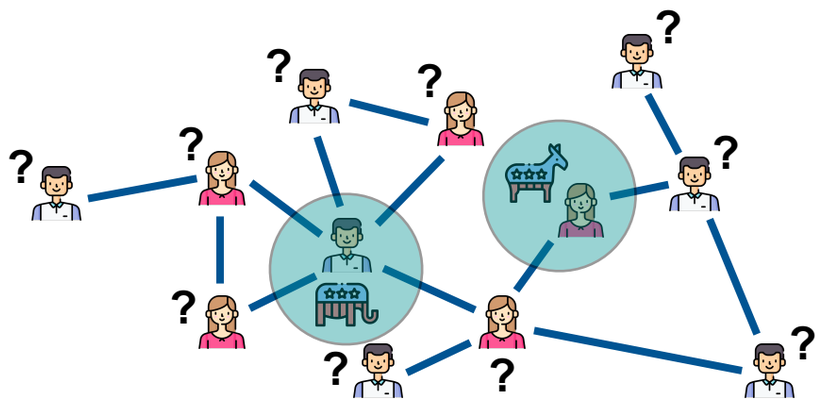


condition?

no condition?

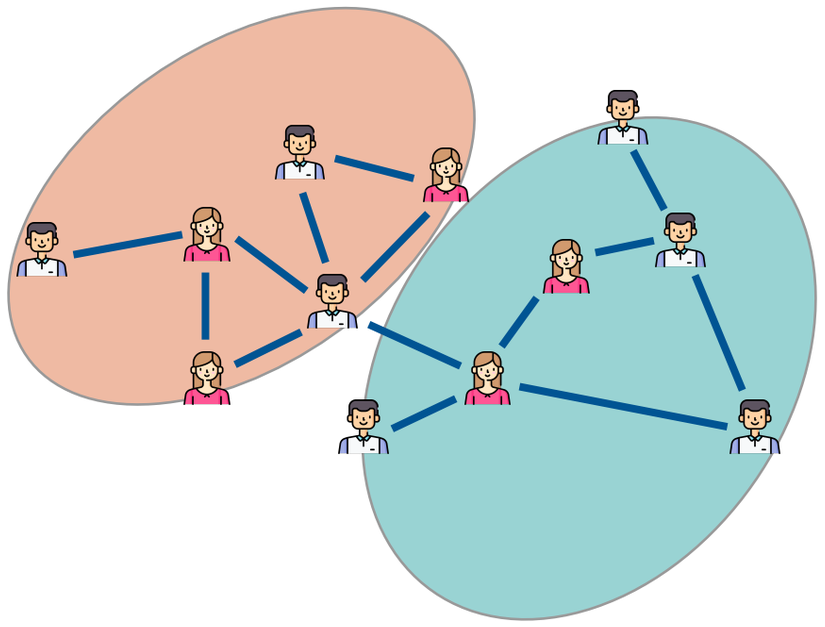
(supervised) graph-level classification

Learning with graph-structured data



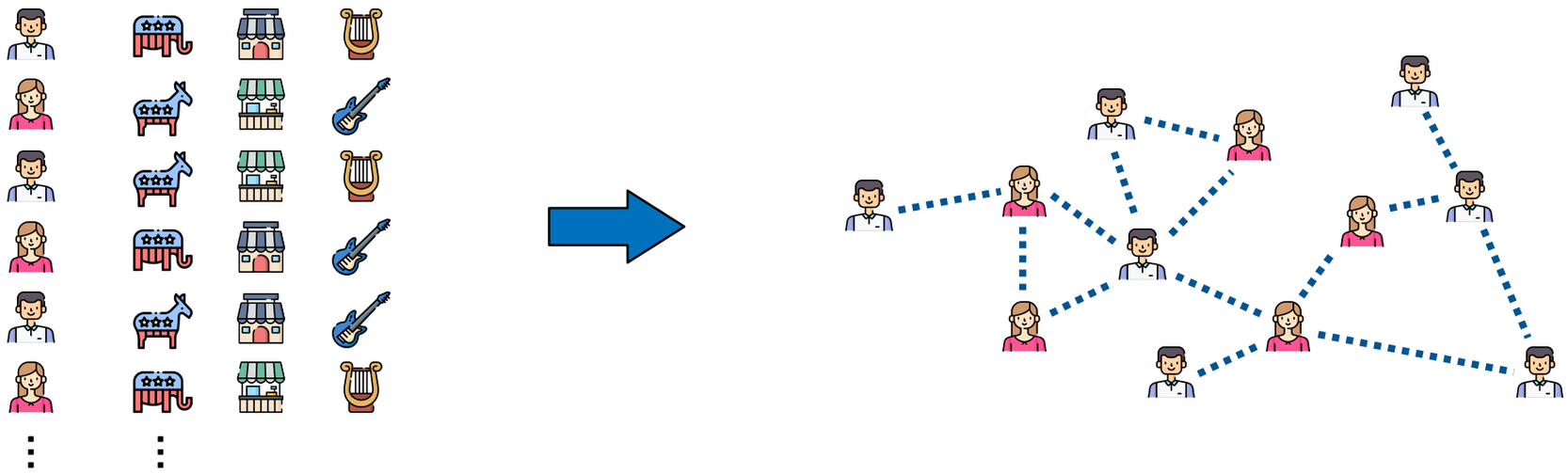
(semi-supervised) node-wise classification

Learning with graph-structured data



(unsupervised) clustering

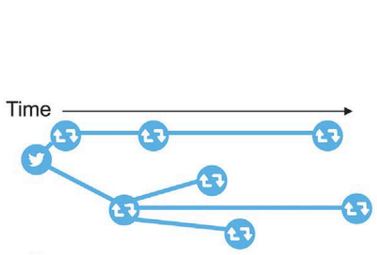
Learning with graph-structured data

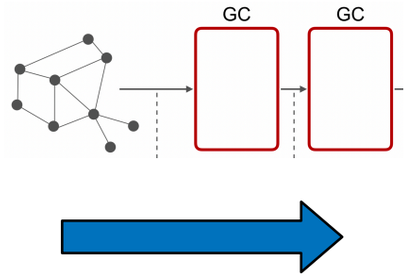


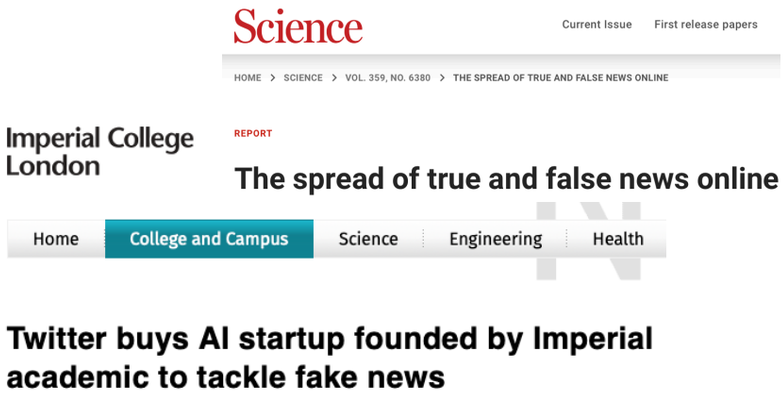
inferring graph topology from data

Exciting possibilities enabled by graph ML

fake news detection



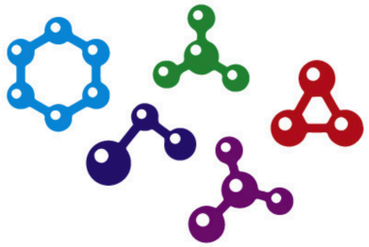


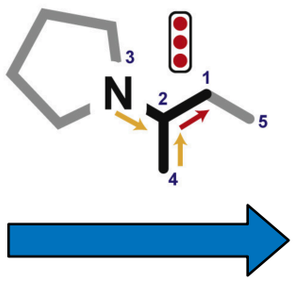


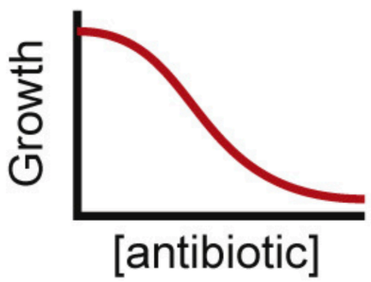
Monti et al., "Fake news detection on social media using geometric deep learning," ICLR Workshop, 2019.

Exciting possibilities enabled by graph ML

drug discovery



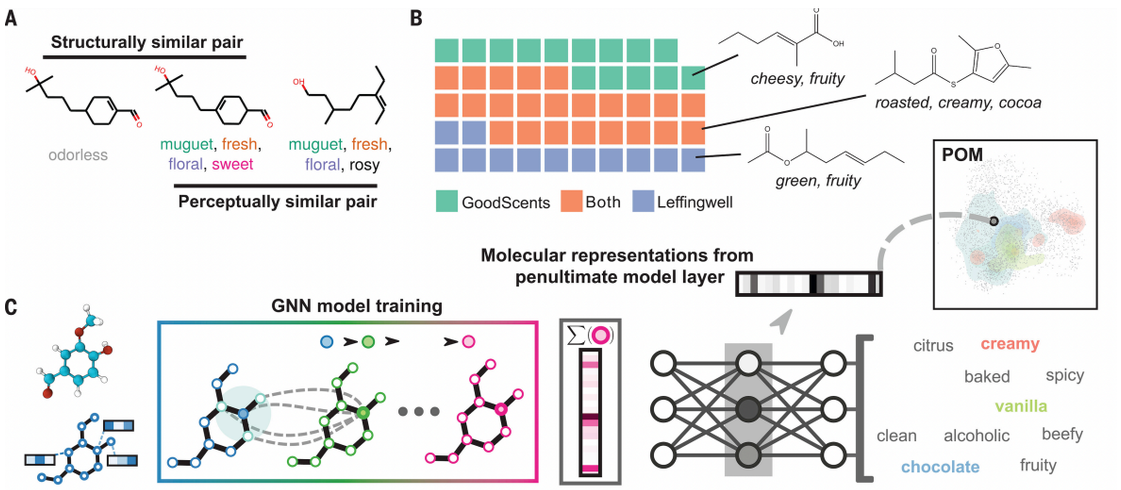




Stokes et al., "A deep learning approach to antibiotic discovery," Cell, 2020.

Exciting possibilities enabled by graph ML

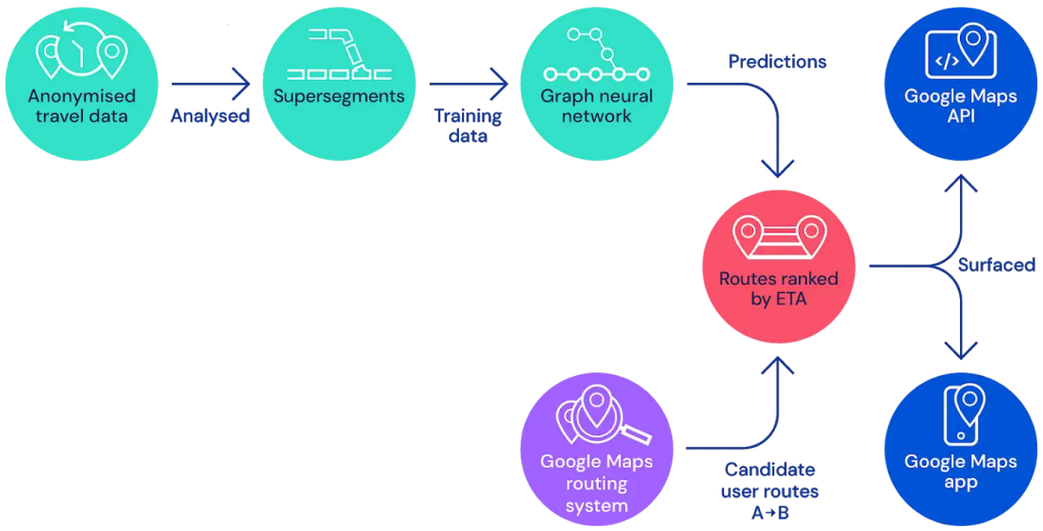
odour perception

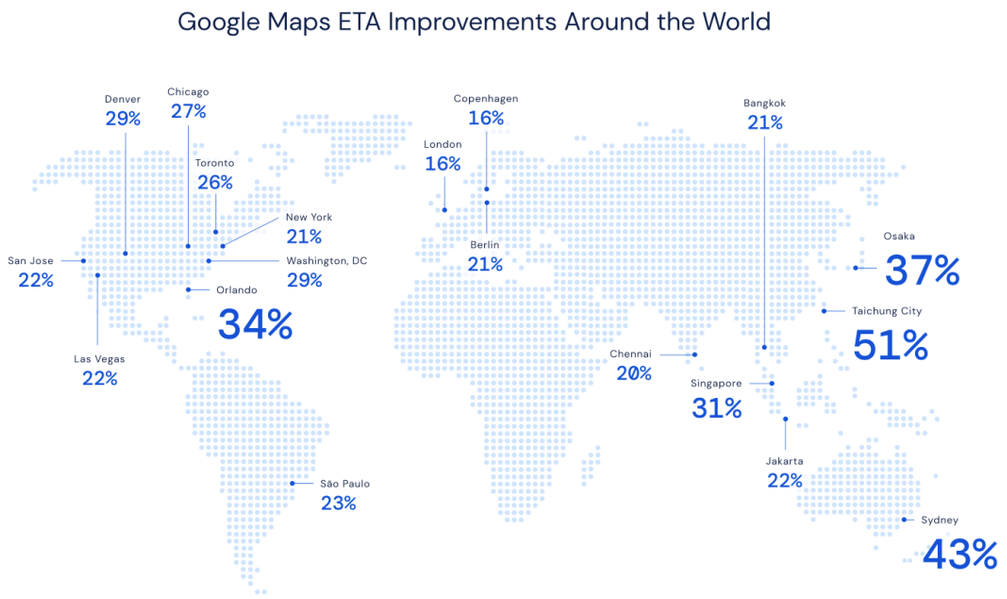


Lee et al., "A principal odor map unifies diverse tasks in olfactory perception," Science, 2023.

Exciting possibilities enabled by graph ML

traffic prediction

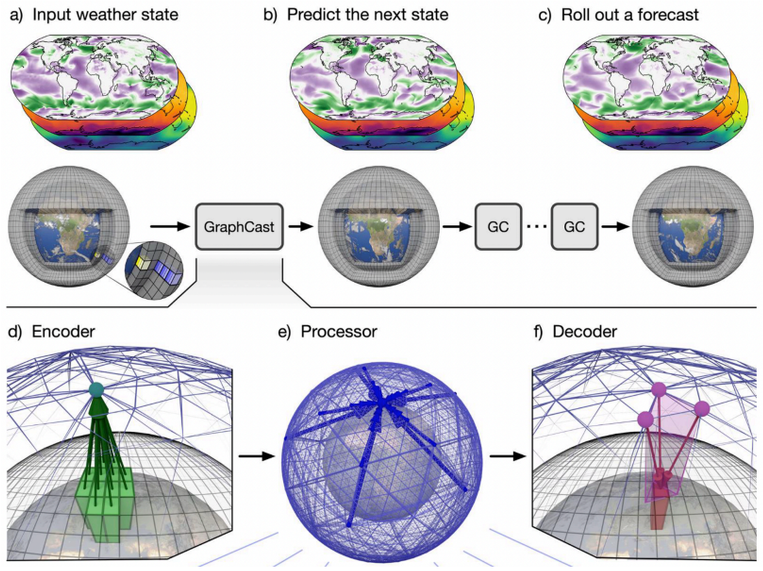




Derrow-Pinion et al., "ETA prediction with graph neural networks in Google Maps," CIKM, 2021.

Exciting possibilities enabled by graph ML

weather forecasting



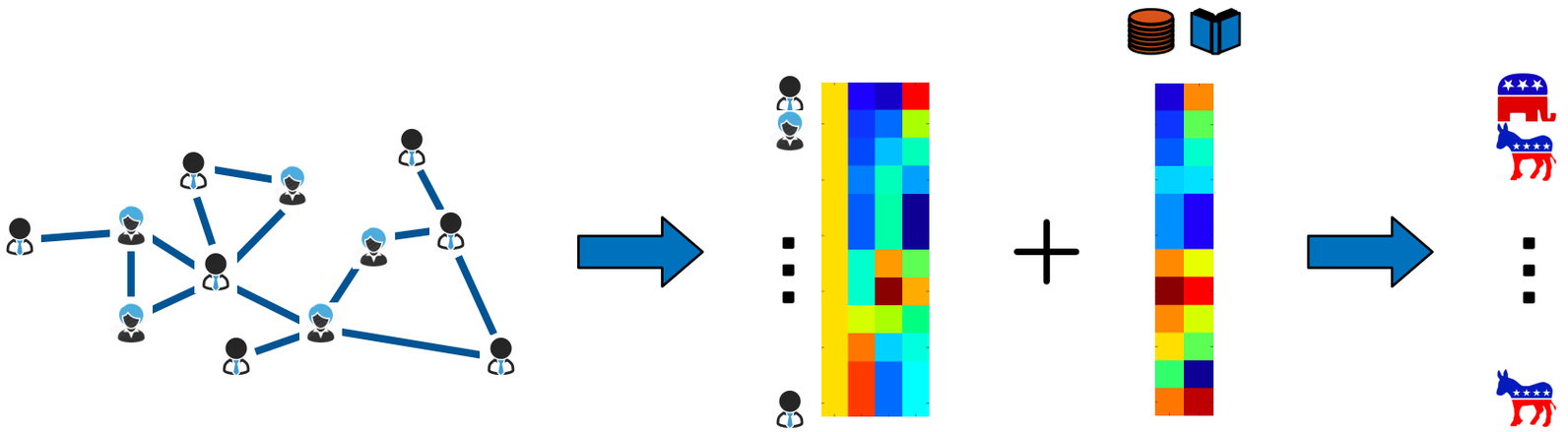
DeepMind AI accurately forecasts weather – on a desktop computer

The machine-learning model takes less than a minute to predict future weather worldwide more precisely than other approaches.

Lam et al., "Learning skillful medium-range global weather forecasting," Science, 2023.

How to incorporate graphs into learning?

- Naïve approach

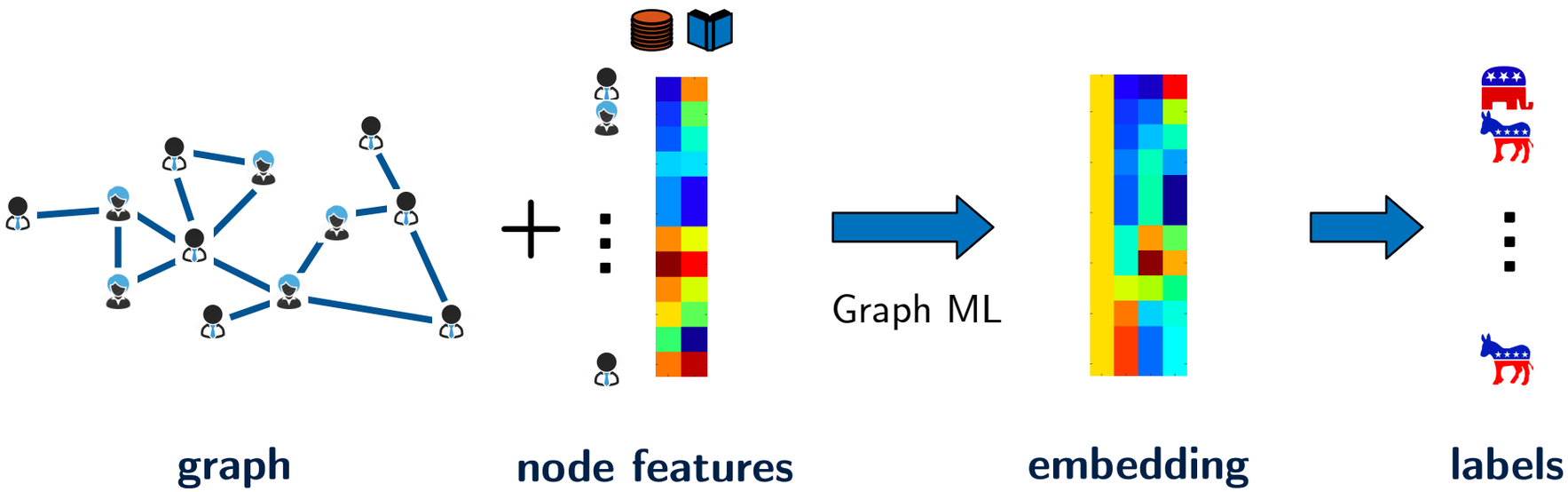


graph **embedding** **node features** **labels**

- Limitations
  - embedding of graph structure leads to information loss
  - (sometimes) computationally expensive

How to incorporate graphs into learning?

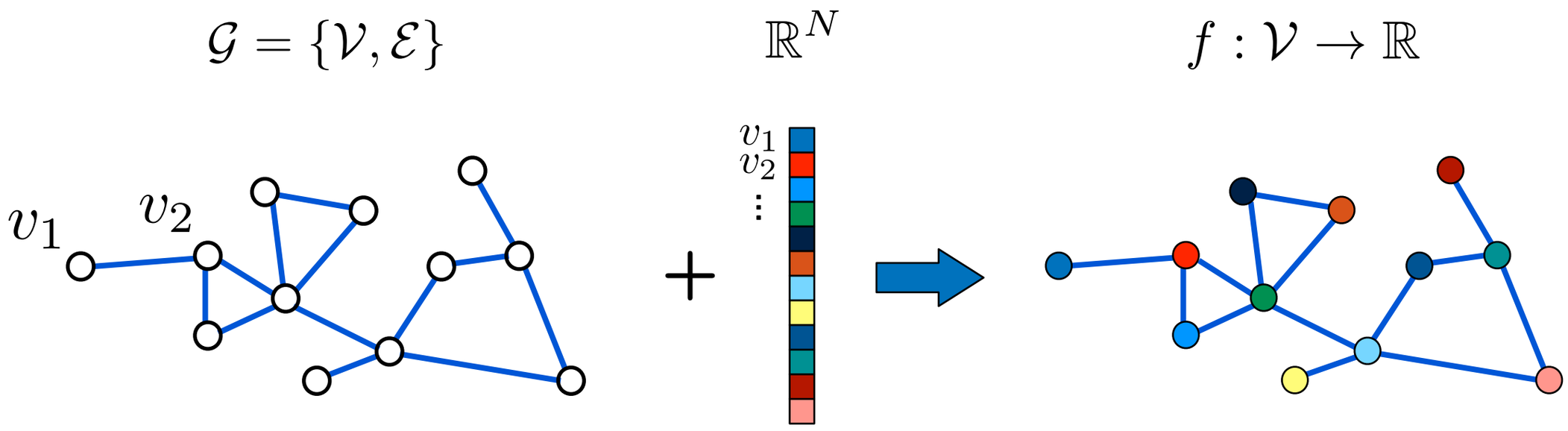
- Idea: directly incorporate graph structure in data analysis



- Need new modelling tools
  - graph signal processing
  - graph neural networks

Graph signal processing

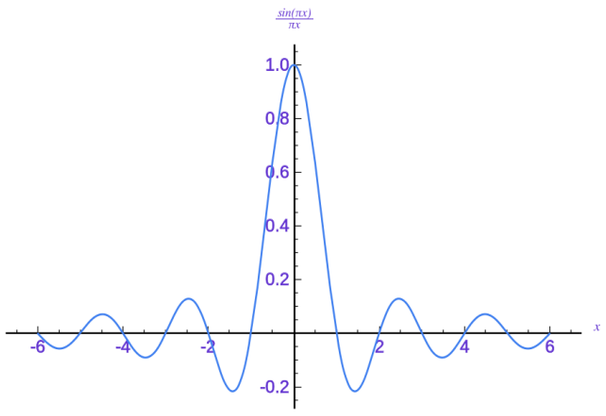
- Graph-structured data can be represented by graph signals



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

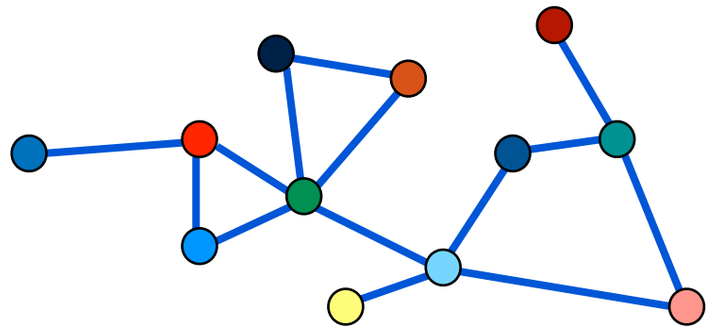
Graph signal processing

1D signal



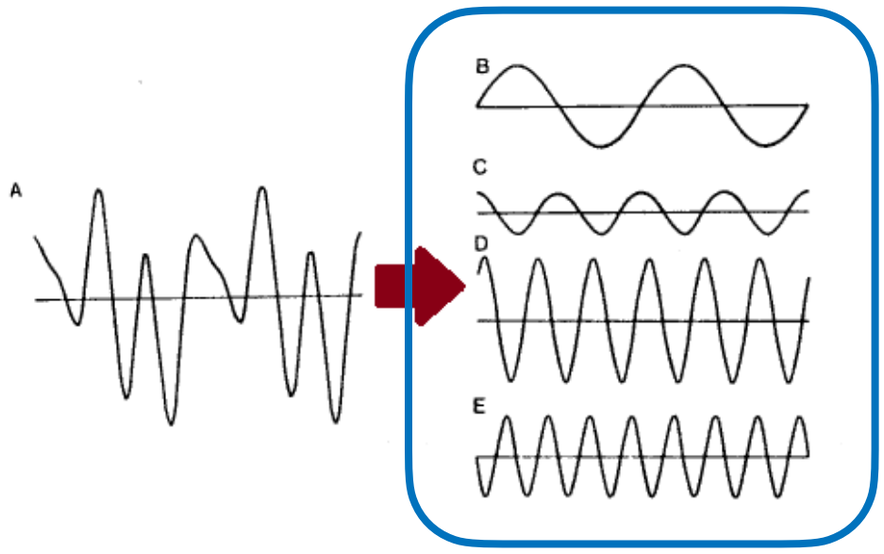
2D signal

$f : \mathcal{V} \to \mathbb{R}$



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

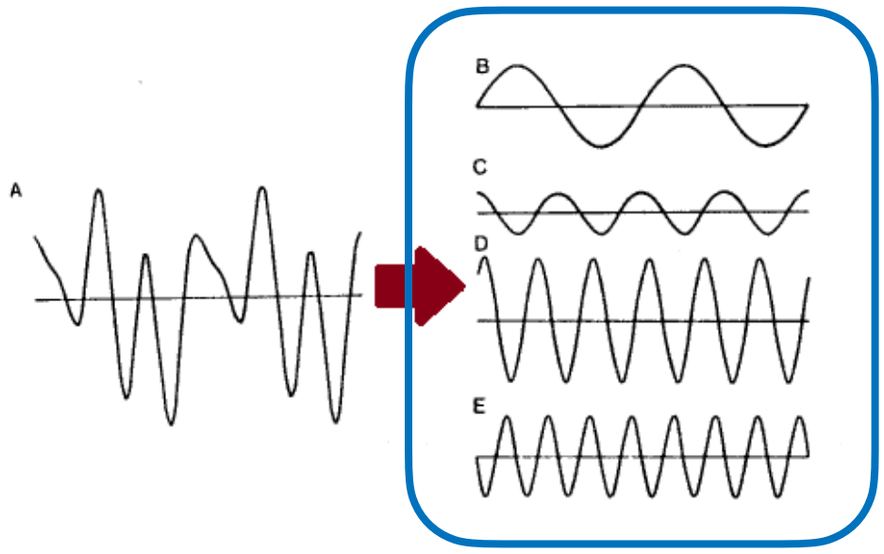
Graph signal processing



classical signal processing

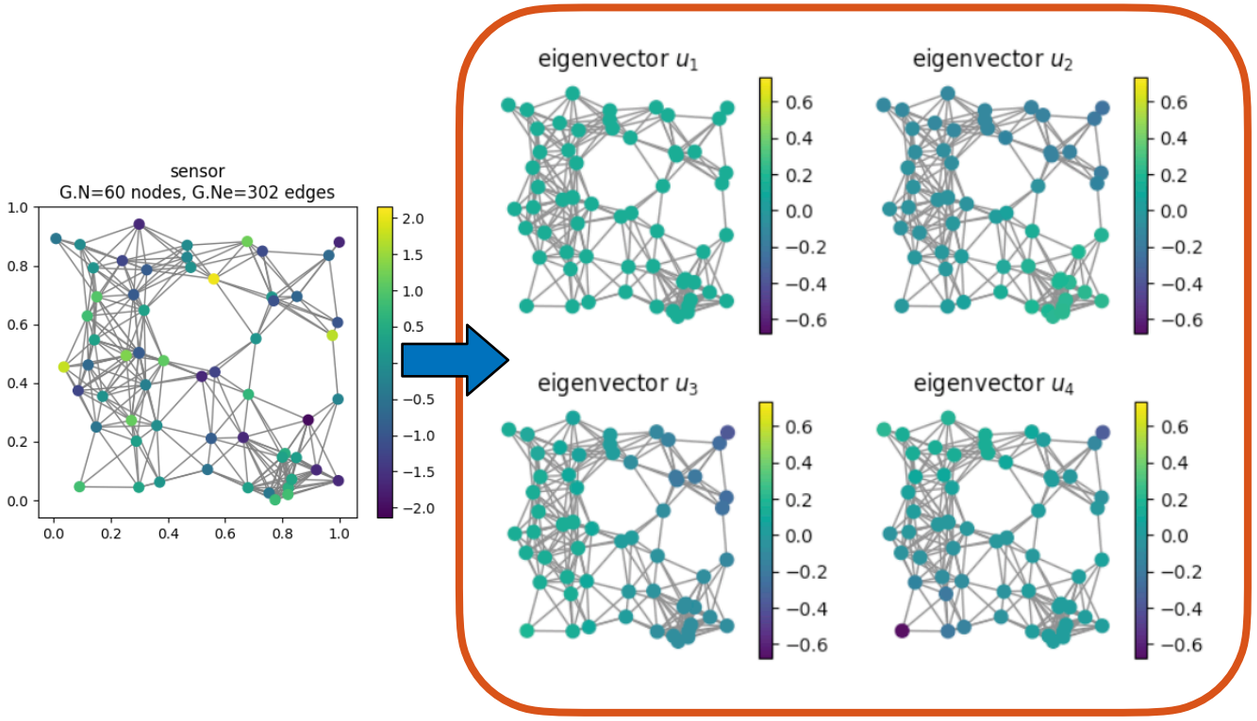
- complex exponentials provide "building blocks" of 1D signal (different oscillations or frequencies)
- leads to **Fourier transform**

Graph signal processing



classical signal processing

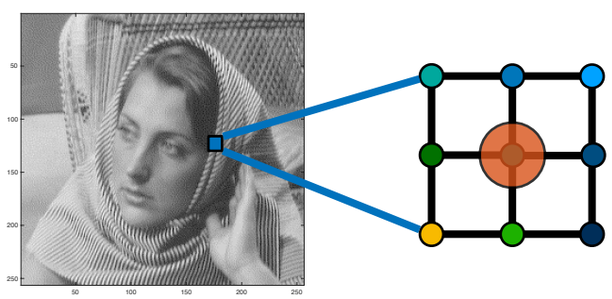
- complex exponentials provide "building blocks" of 1D signal (different oscillations or frequencies)
- leads to **Fourier transform**

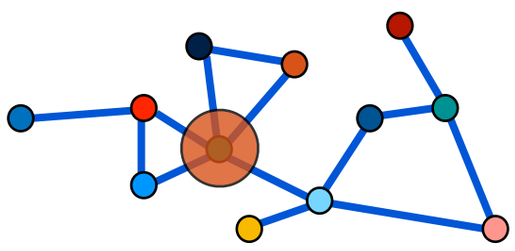


graph signal processing

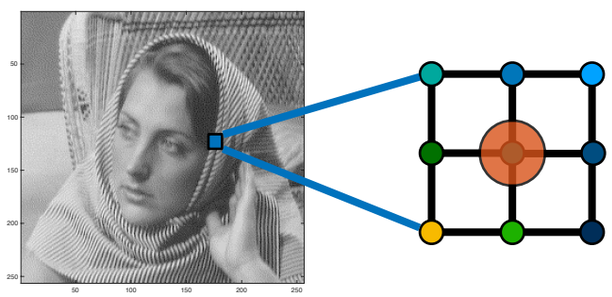
- Laplacian eigenvectors provide "building blocks" of graph signal (different oscillation or frequencies)
- leads to **graph Fourier transform**
- enables **convolution** and **filtering** on graphs

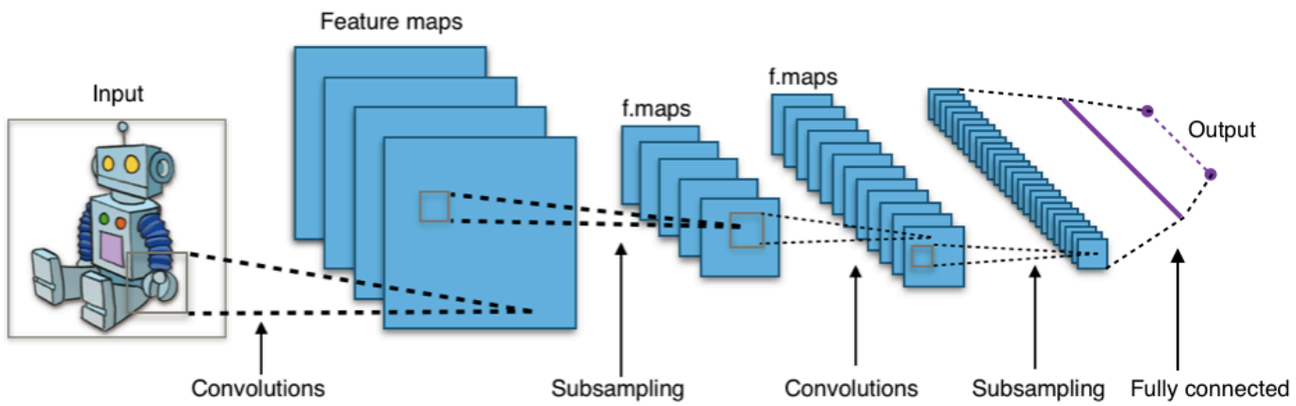
Convolutional neural networks on graphs

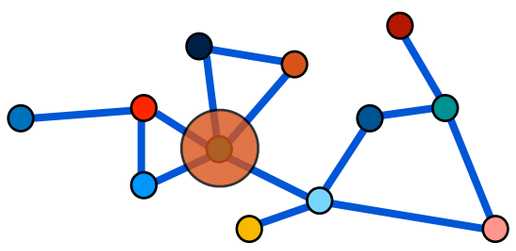


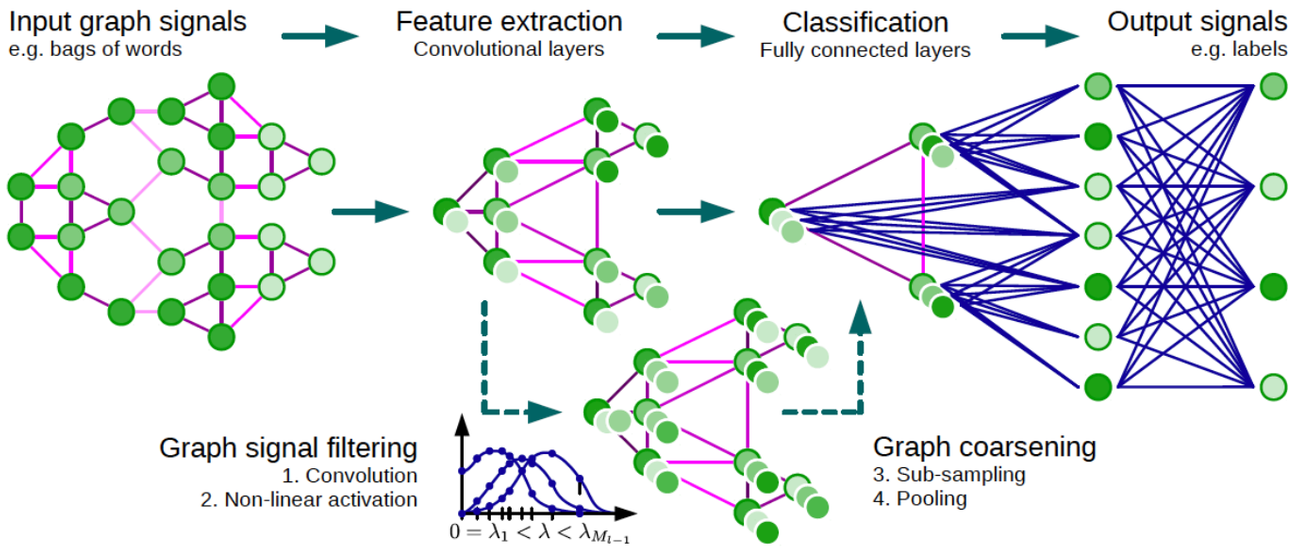


Convolutional neural networks on graphs

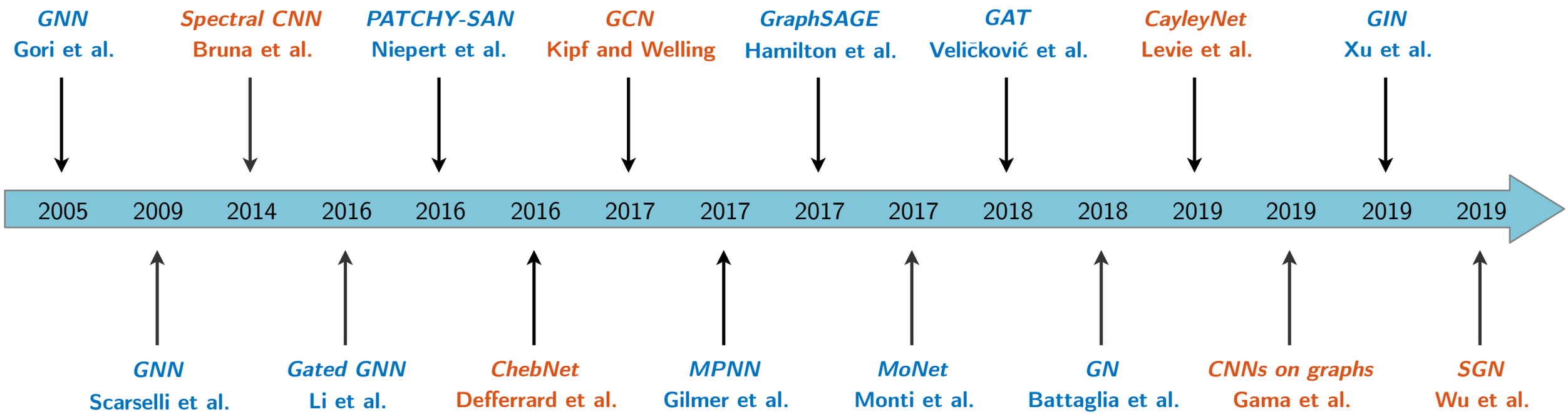






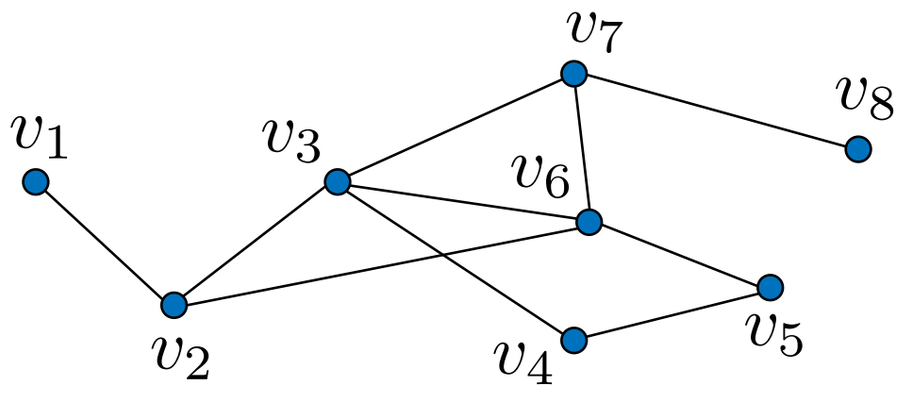


(More generally) Graph neural networks



Graph Signal Processing

Graphs and graph Laplacian



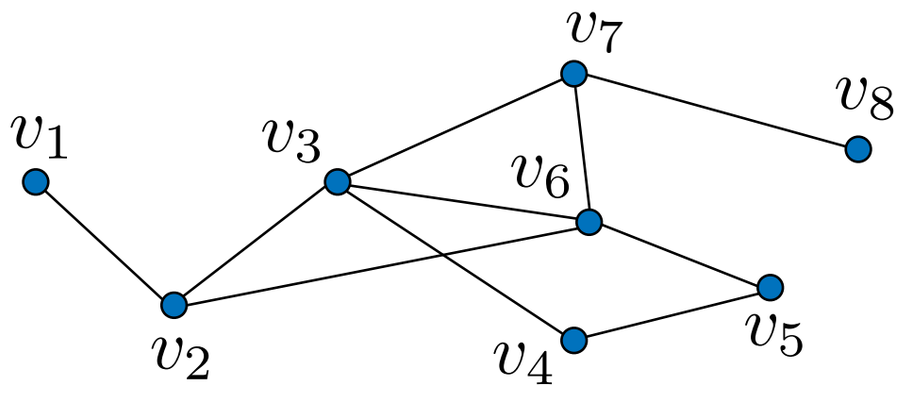
weighted and undirected graph:

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

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

W

Graphs and graph Laplacian



weighted and undirected graph:

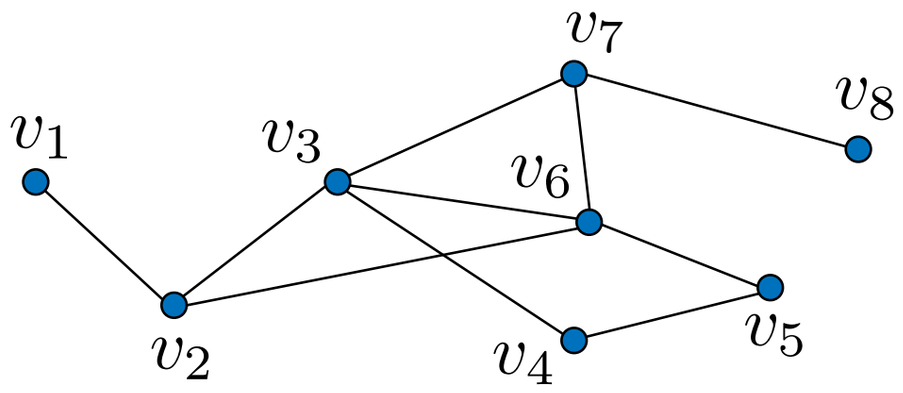
$$\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$$

$$D = \mathrm{diag}(d(v_1), \cdots, d(v_N))$$

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

D W

Graphs and graph Laplacian



weighted and undirected graph:

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

$$D = \mathrm{diag}(d(v_1), \cdots, d(v_N))$$

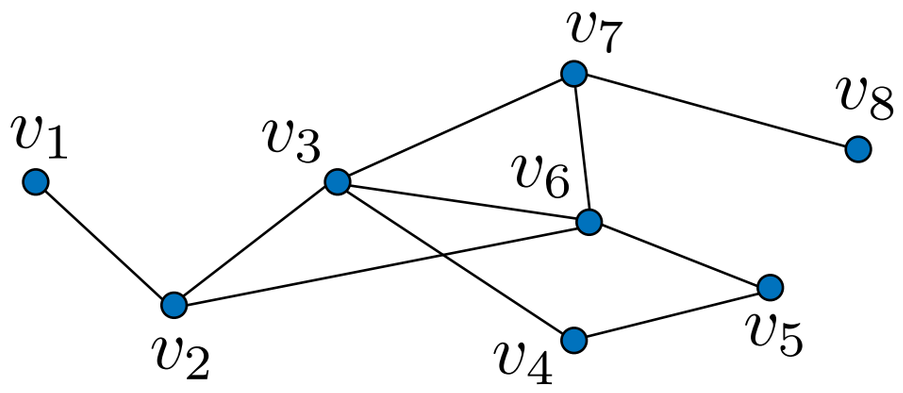
$$L = D - W \qquad \textbf{equivalent to G!}$$

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

D W L

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

Graphs and graph Laplacian



weighted and undirected graph:

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

$$D = \mathrm{diag}(d(v_1), \cdots, d(v_N))$$

$$L = D - W \qquad \textbf{equivalent to G!}$$

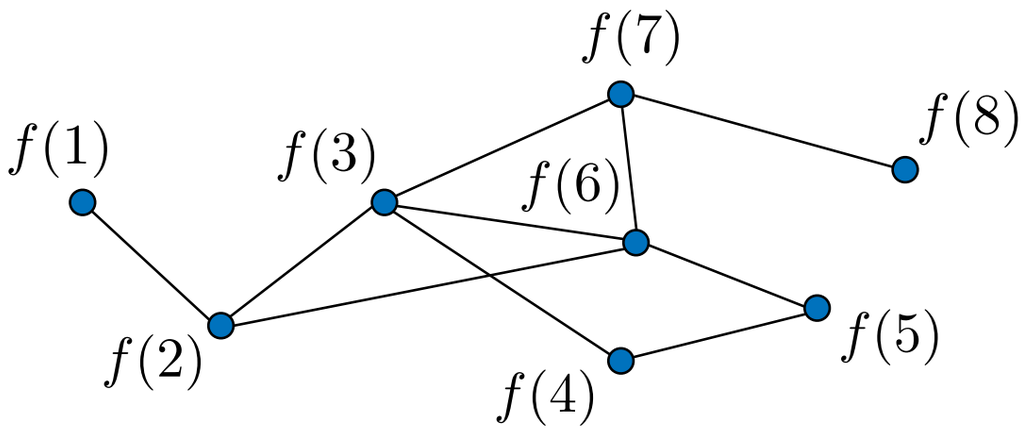
$$L_{\mathrm{norm}} = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}$$

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

D W L

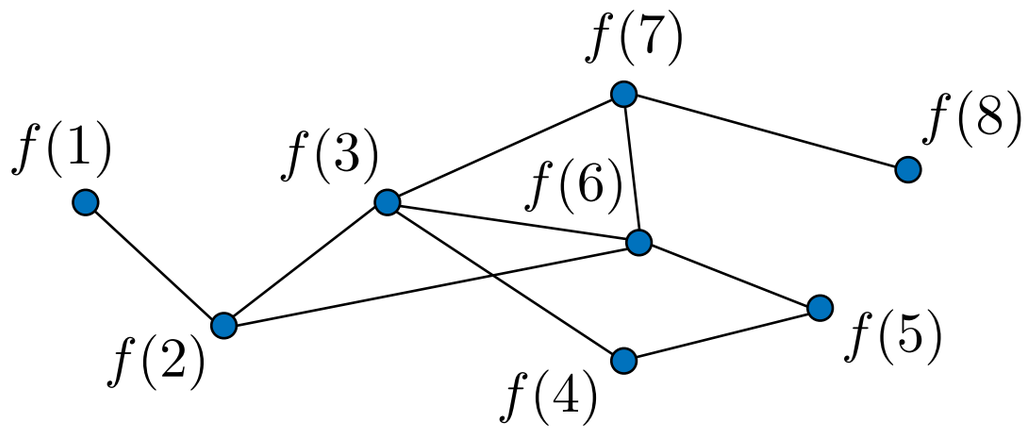
- symmetric
- off-diagonal entries non-positive
- rows sum up to zero

Graph Laplacian



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

Graph Laplacian

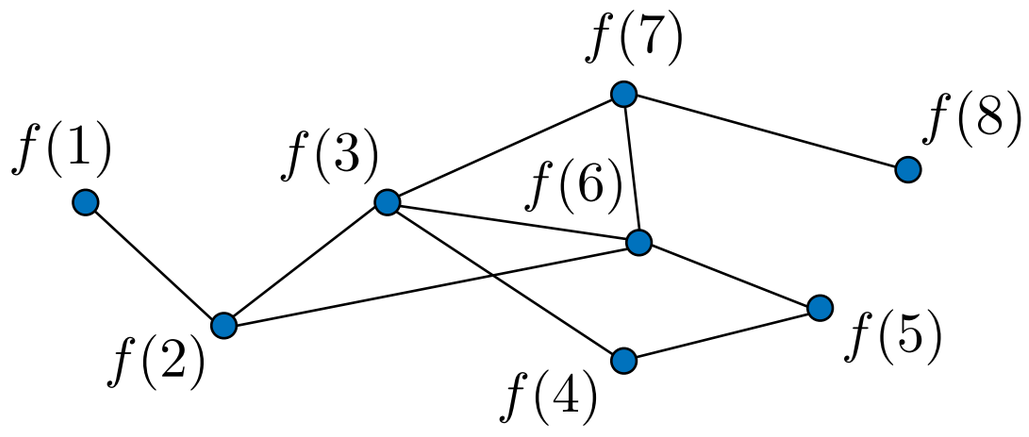


graph signal $f : \mathcal{V} \to \mathbb{R}$

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

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

Graph Laplacian



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

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

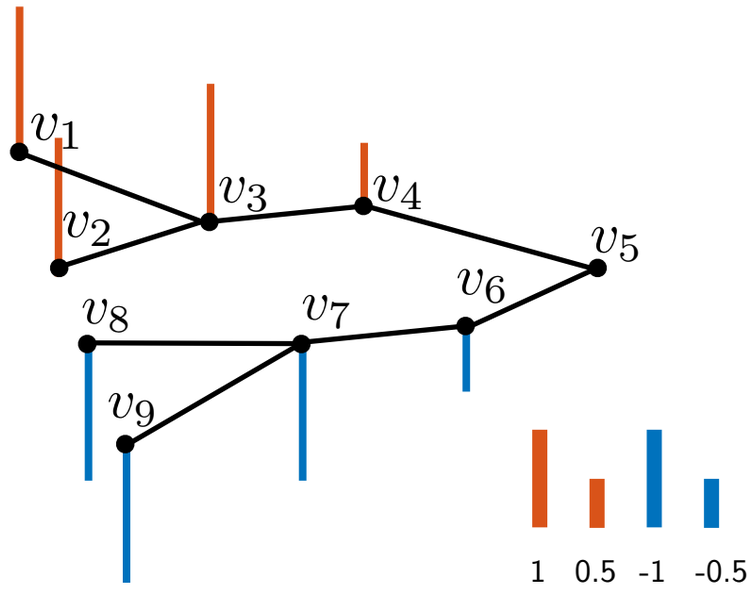
$$\begin{pmatrix} f(1) \\ f(2) \\ f(3) \\ f(4) \\ f(5) \\ f(6) \\ f(7) \\ f(8) \end{pmatrix}^T \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & -1 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & -1 &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

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

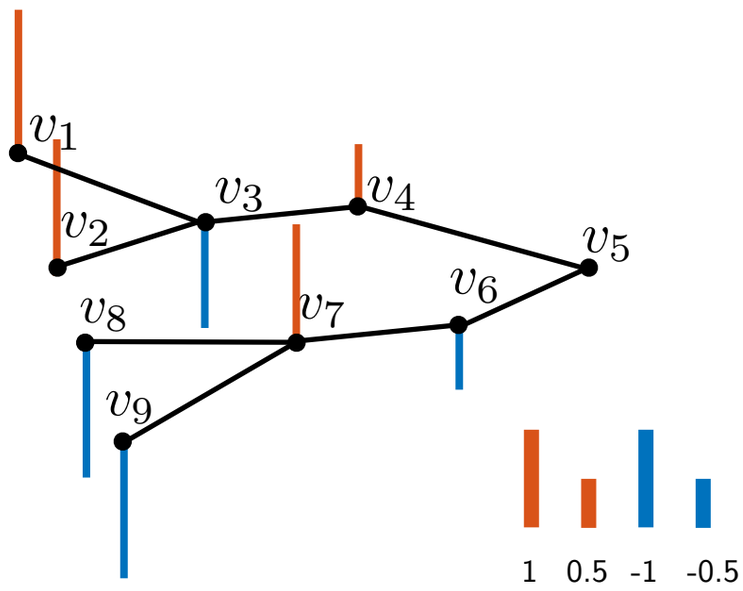
$$f^T L f = \frac{1}{2} \sum_{i,j=1}^{N} W_{ij} \left(f(i) - f(j)\right)^2$$

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

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

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

Graph Laplacian

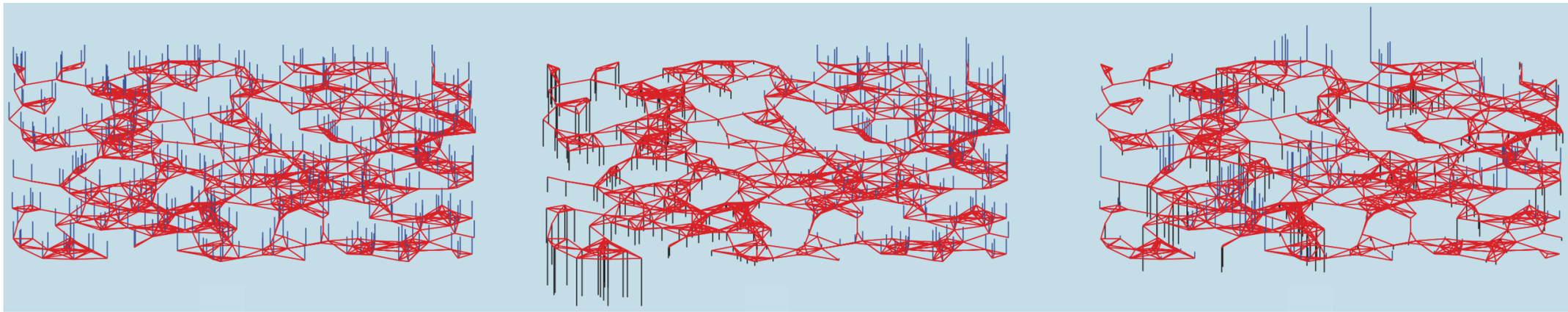
- L has a complete set of orthonormal eigenvectors: $L = \chi \Lambda \chi^T$

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

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

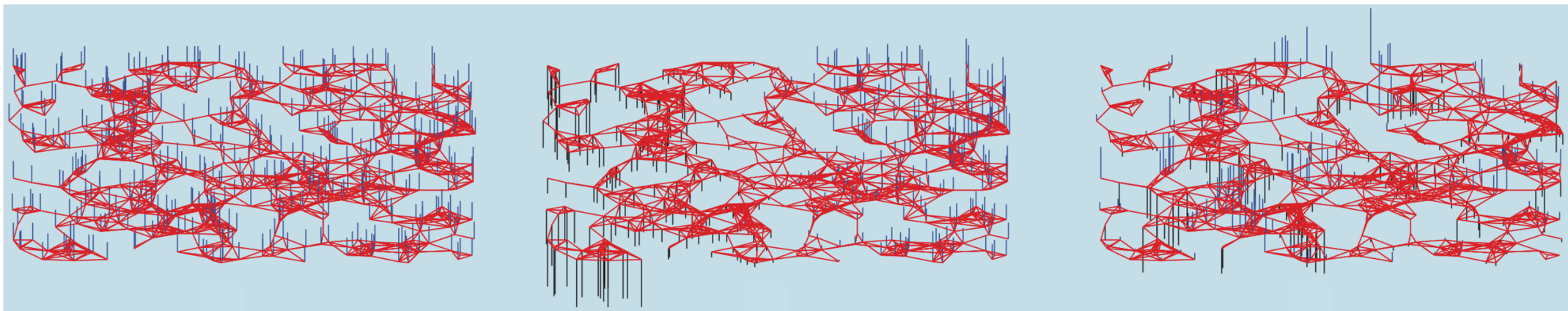
- Eigenvalues are usually sorted increasingly: $0 = \lambda_0 < \lambda_1 \leq \ldots \leq \lambda_{N-1}$

Graph Fourier transform



χ_0 χ_1 χ_{50}

Graph Fourier transform



χ_0 χ_1 χ_{50}

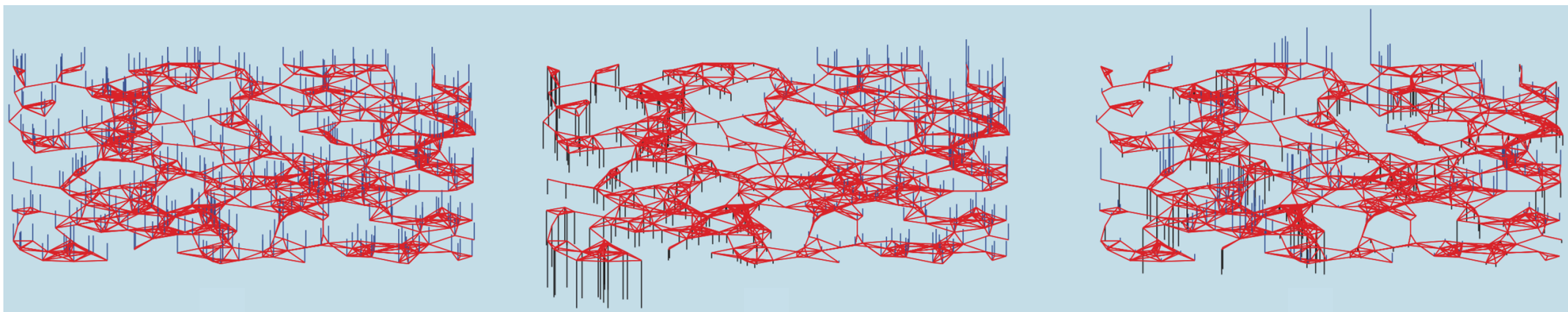
low frequency → high frequency

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

$\chi_0^T L \chi_0 = \lambda_0 = 0$ $\chi_{50}^T L \chi_{50} = \lambda_{50}$

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

Graph Fourier transform



χ_0 χ_1 χ_{50}

low frequency → high frequency

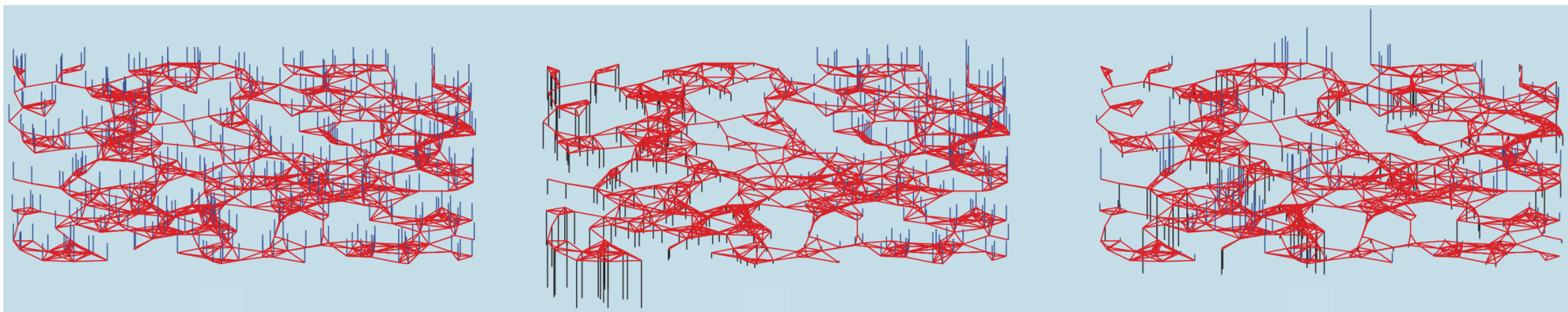
$L = \chi \Lambda \chi^T$

$\chi_0^T L \chi_0 = \lambda_0 = 0$ $\chi_{50}^T L \chi_{50} = \lambda_{50}$

graph Fourier transform:

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

Graph Fourier transform



χ_0 χ_1 χ_{50}

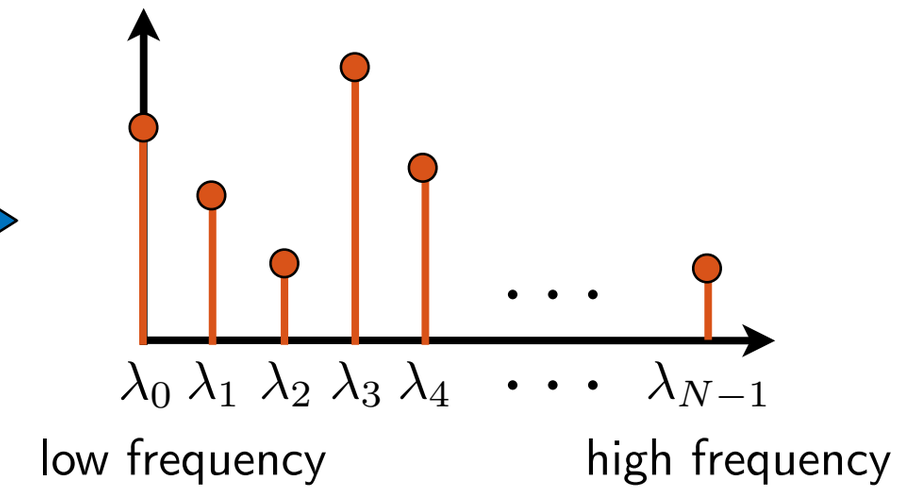
low frequency → high frequency

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

$\chi_0^T L \chi_0 = \lambda_0 = 0$ $\chi_{50}^T L \chi_{50} = \lambda_{50}$

graph Fourier transform:

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



Graph Fourier transform

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

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 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: χ_ℓ

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

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

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

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: χ_ℓ

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

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

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

Graph Fourier transform

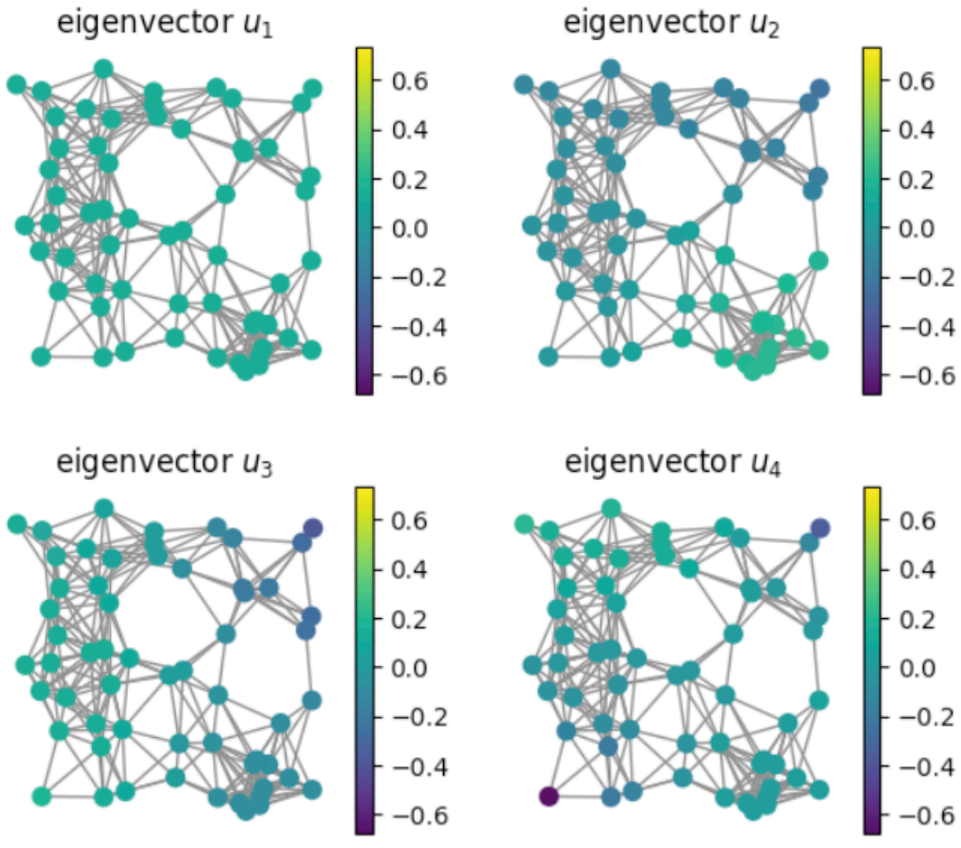
- Graph Fourier transform

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

Graph Fourier transform

- Graph Fourier transform

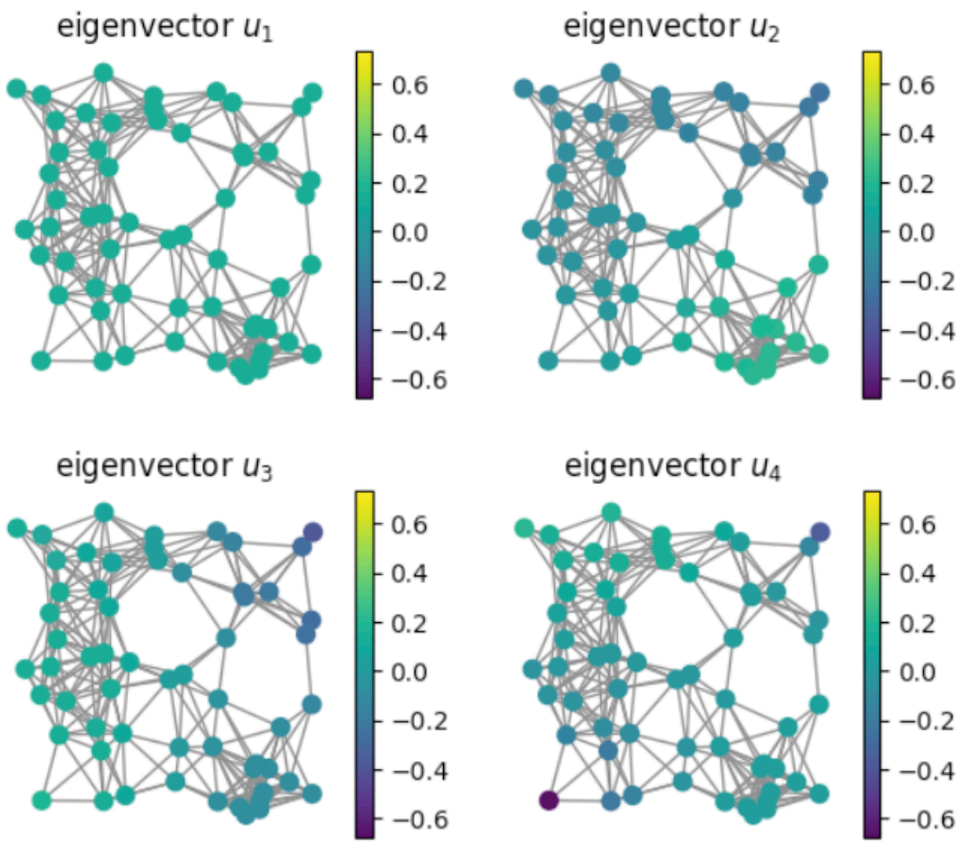
$$\hat{f}(\ell) = \langle \chi_\ell, f \rangle : \begin{bmatrix} | & & | \\ \chi_0 & \cdots & \chi_{N\text{-}1} \\ | & & | \end{bmatrix}^T \begin{matrix} | \\ f \\ | \end{matrix}$$

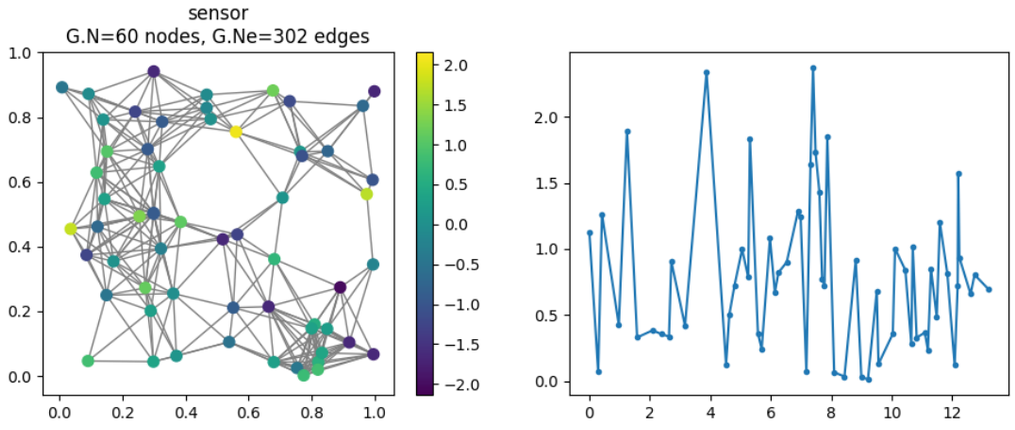


Graph Fourier transform

- Graph Fourier transform

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

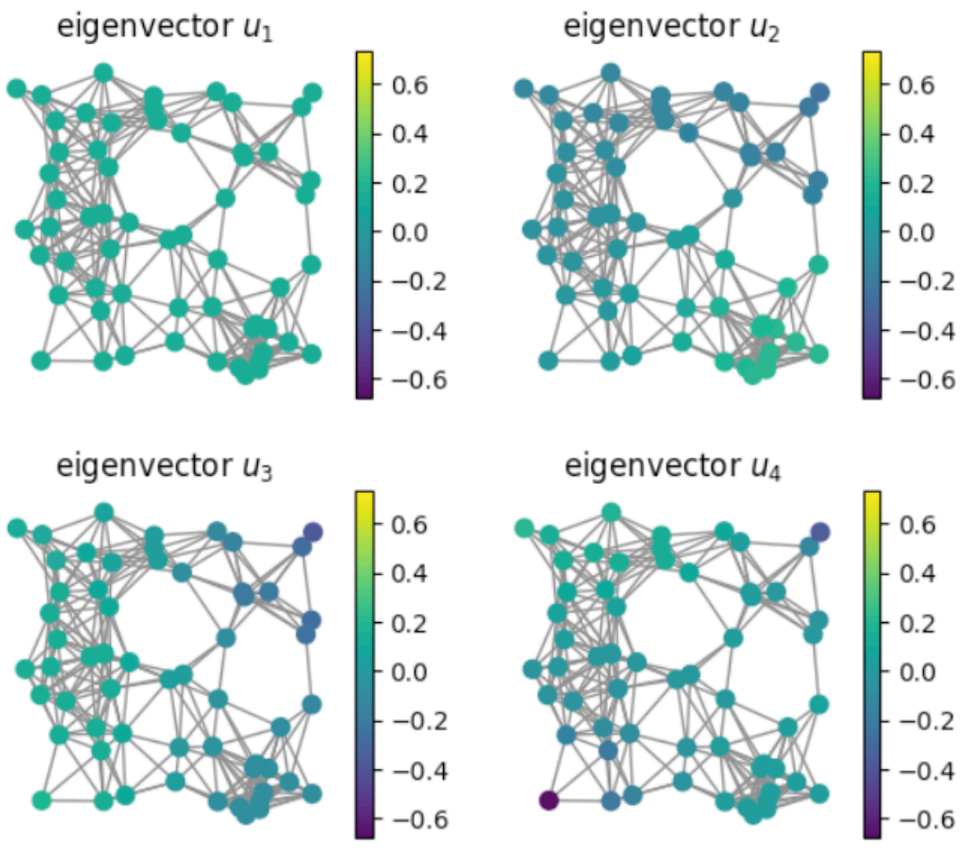


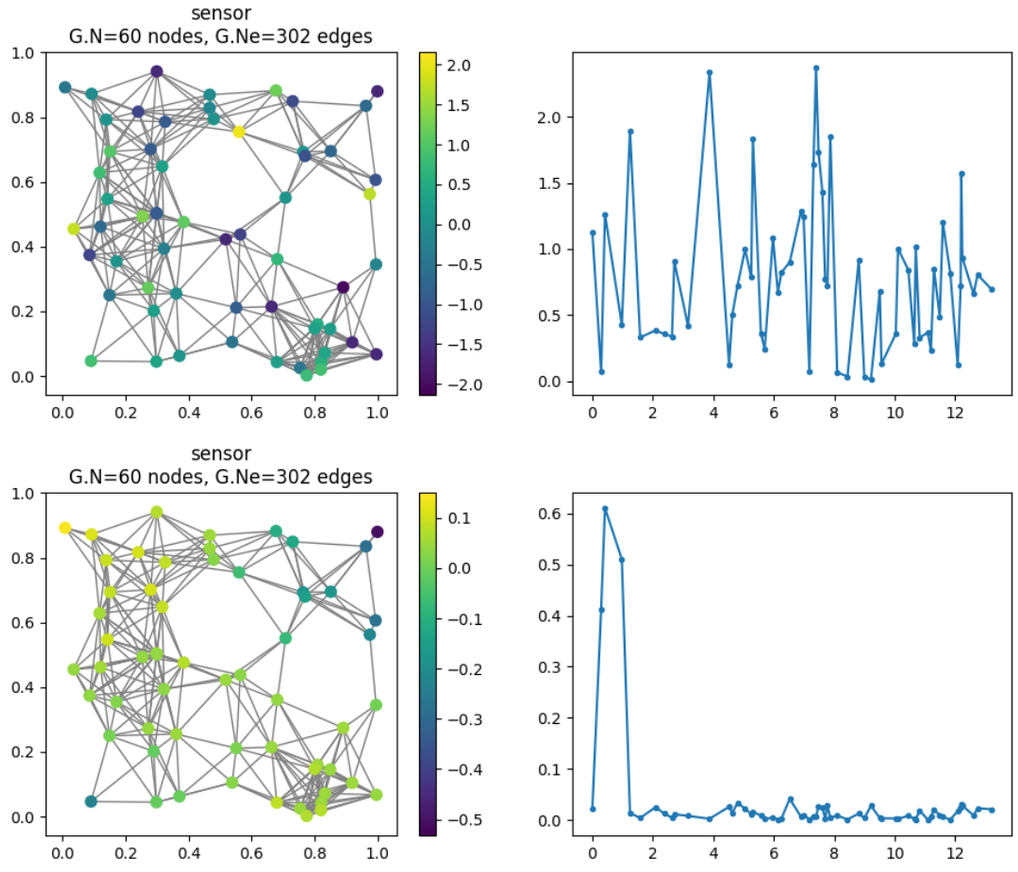


Graph Fourier transform

- Graph Fourier transform

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





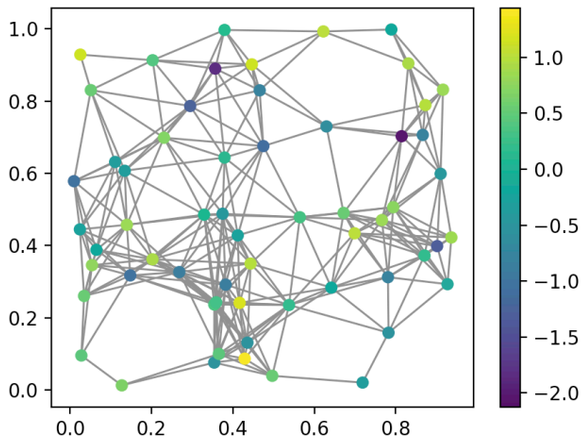
Graph spectral filtering

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 spectral filtering

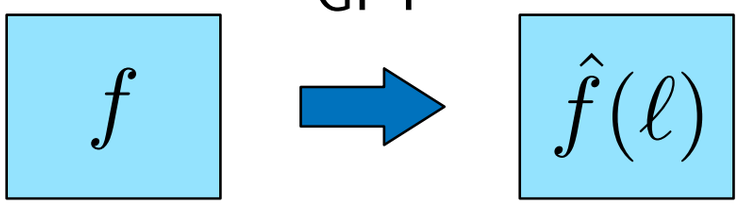
GFT: $\hat{f}(\ell) = \langle \chi_\ell, f \rangle = \sum_{i=1}^{N} \chi_\ell^*(i) f(i)$ $\quad f(i) = \sum_{\ell=0}^{N-1} \hat{f}(\ell) \chi_\ell(i)$

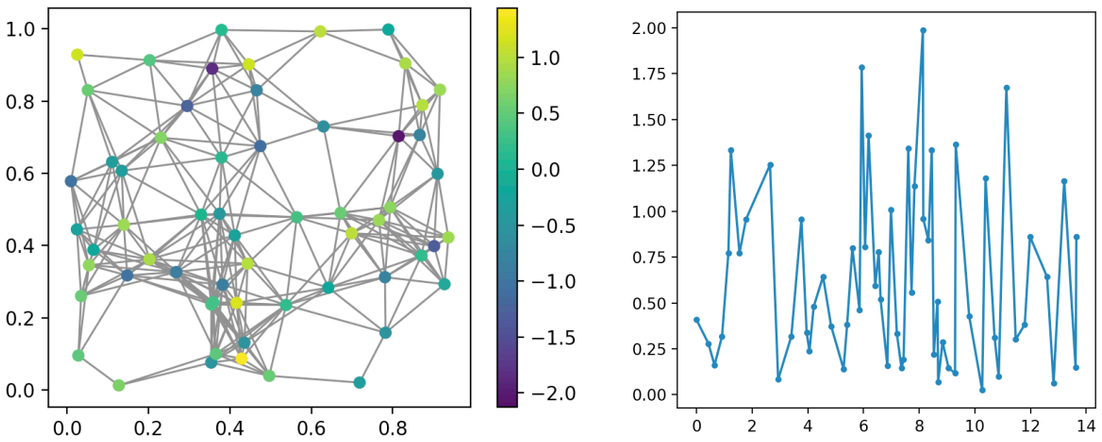
f



Graph spectral filtering

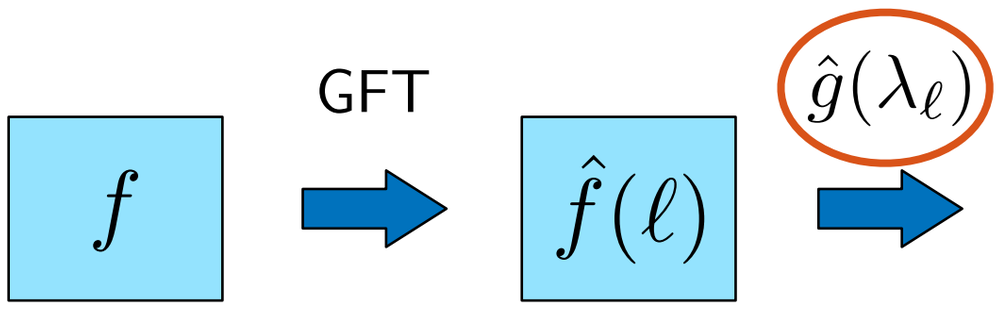
GFT: $\hat{f}(\ell) = \langle \chi_\ell, f \rangle = \sum_{i=1}^{N} \chi_\ell^*(i) f(i)$ $\quad f(i) = \sum_{\ell=0}^{N-1} \hat{f}(\ell) \chi_\ell(i)$

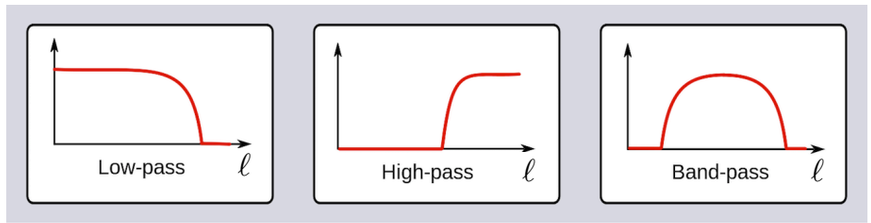


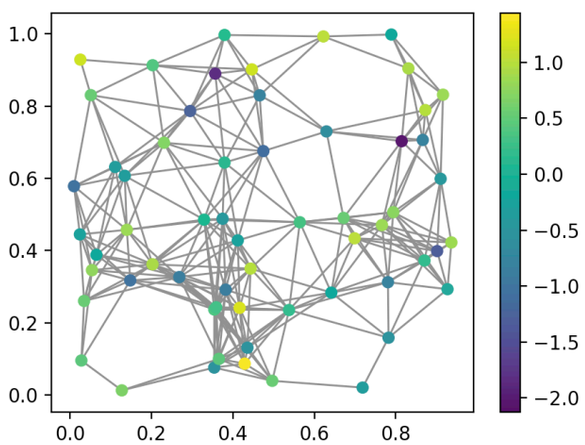


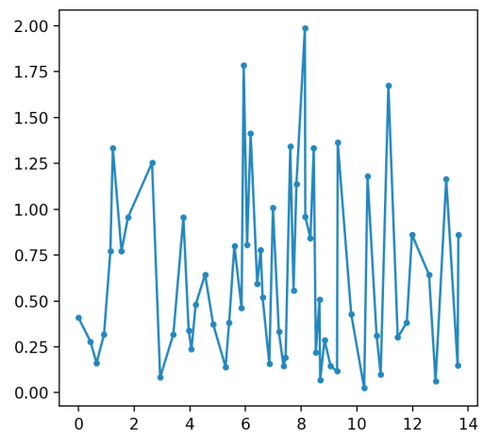
Graph spectral filtering

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



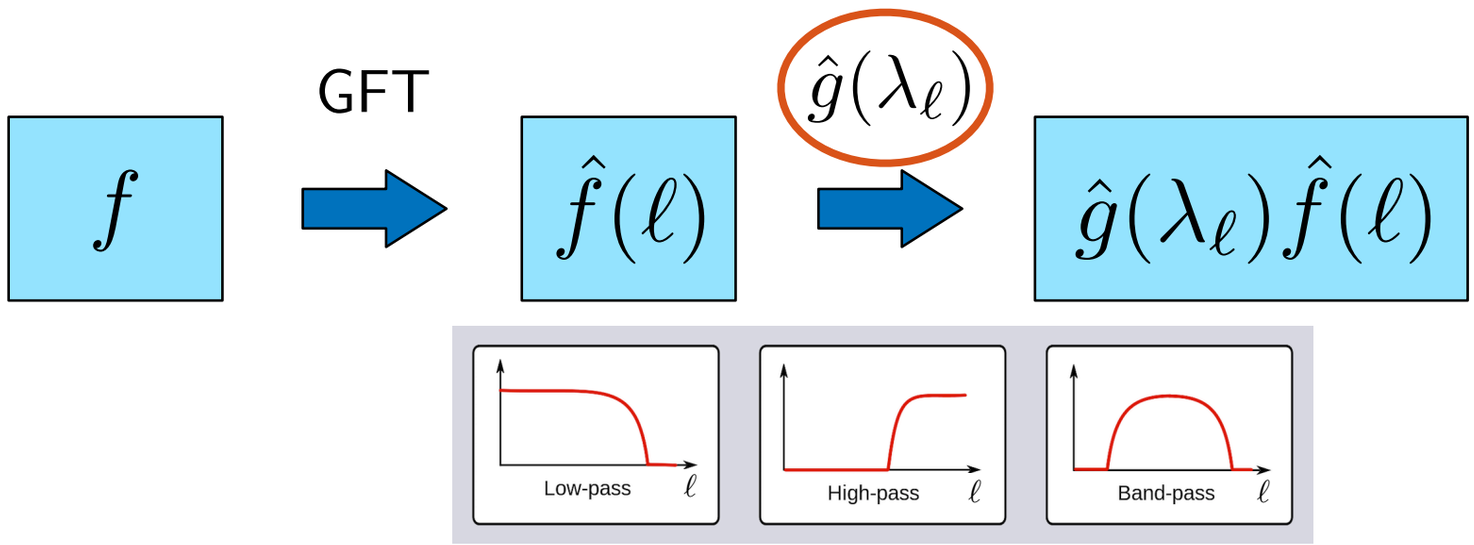


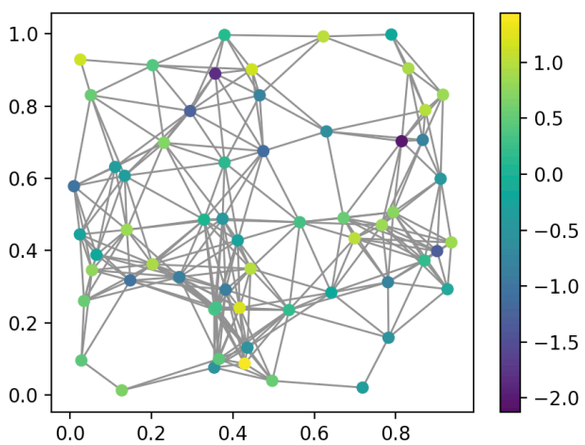


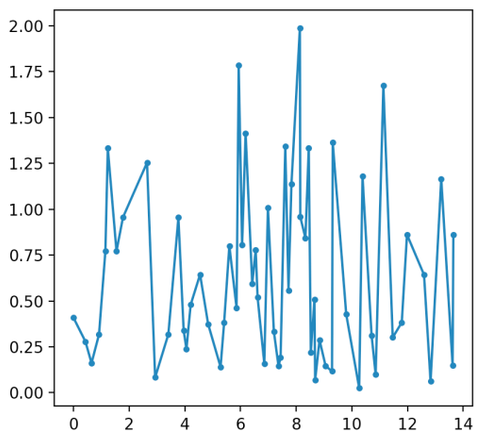


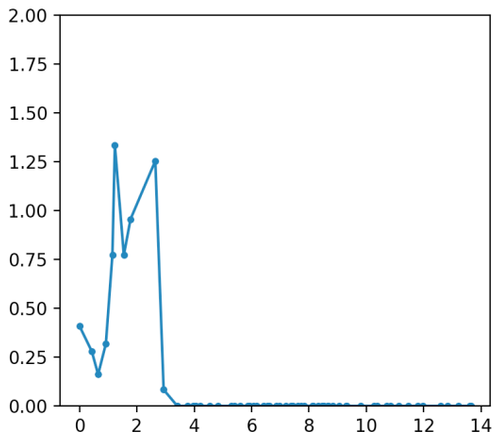
Graph spectral filtering

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



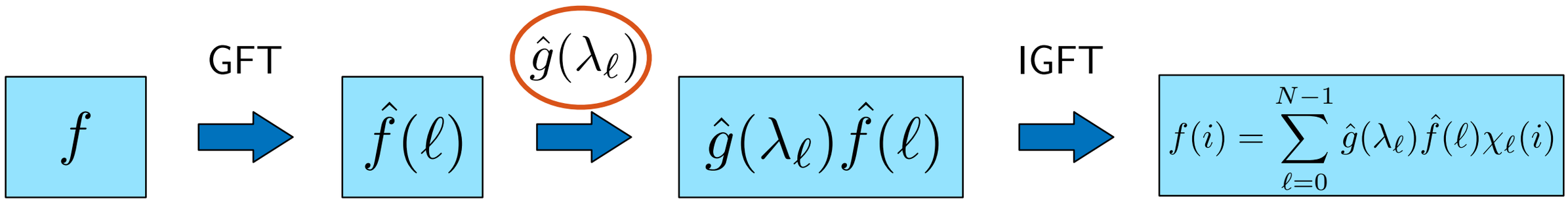


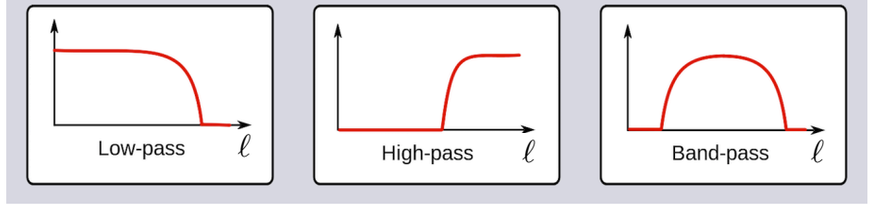


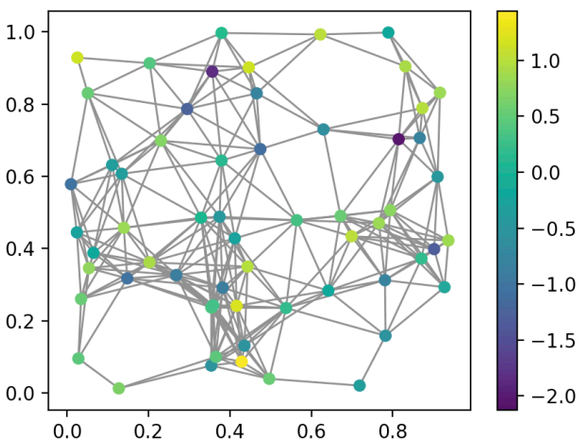


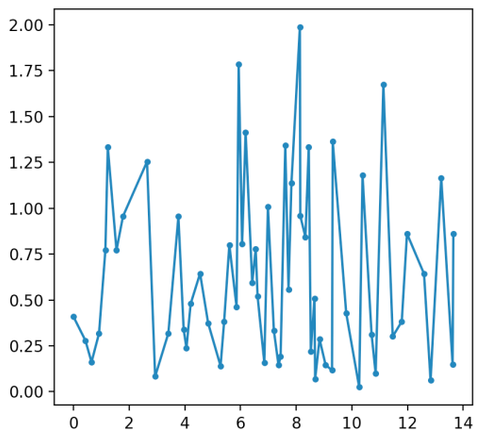
Graph spectral filtering

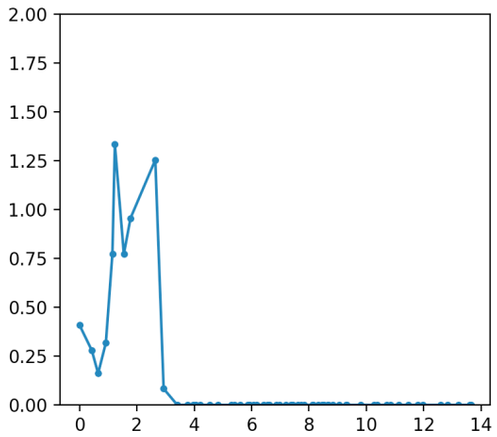
GFT: $\hat{f}(\ell) = \langle \chi_\ell, f \rangle = \sum_{i=1}^{N} \chi_\ell^*(i) f(i)$ $\quad f(i) = \sum_{\ell=0}^{N-1} \hat{f}(\ell) \chi_\ell(i)$

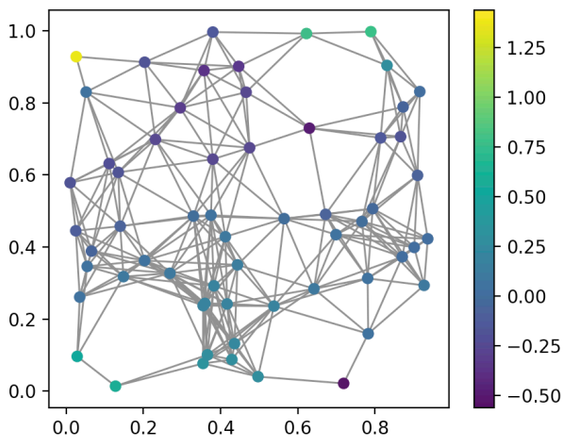




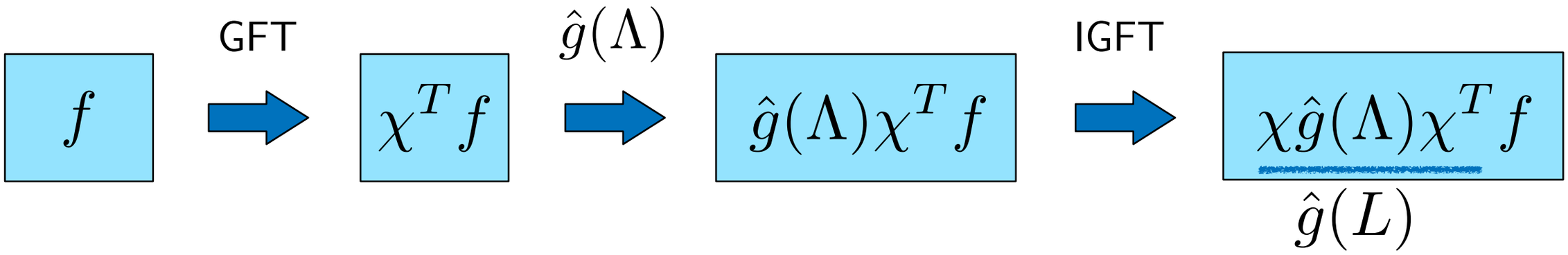




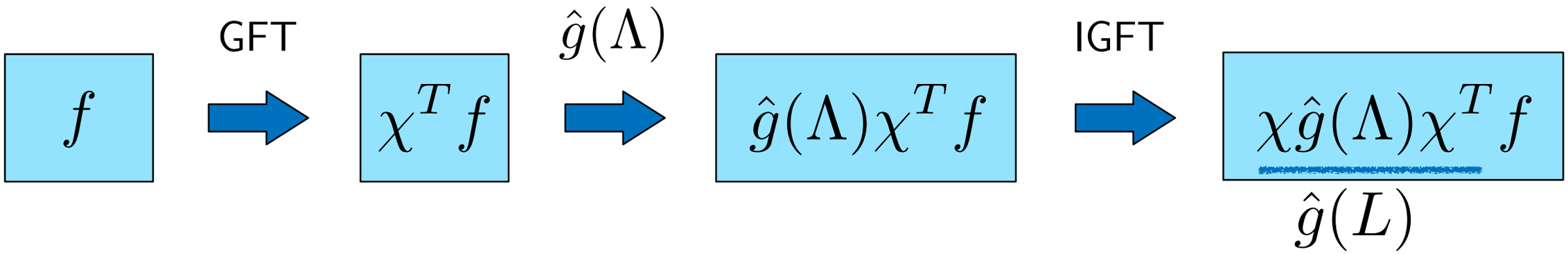




A practical example



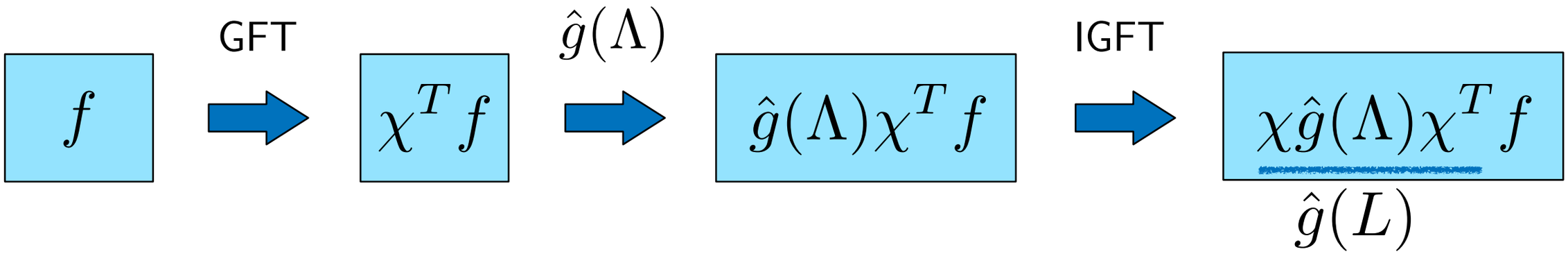
A practical example



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

$$y^* = \arg\min_y \{||y - f||_2^2 + \gamma y^T L y\}$$

A practical example



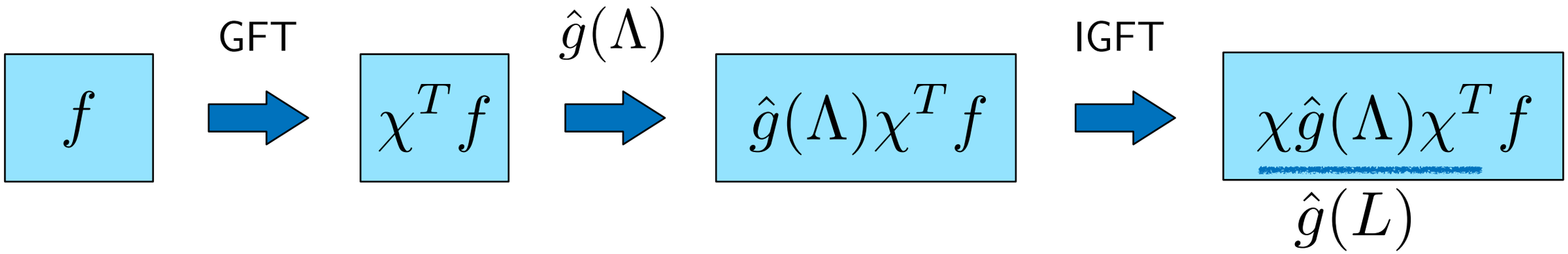
problem: we observe a noisy graph signal $f = y_0 + \eta$ and wish to recover y_0

$$y^* = \arg\min_y \{||y - f||_2^2 + \gamma y^T L y\}$$

data fitting term

"smoothness" assumption

A practical example



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

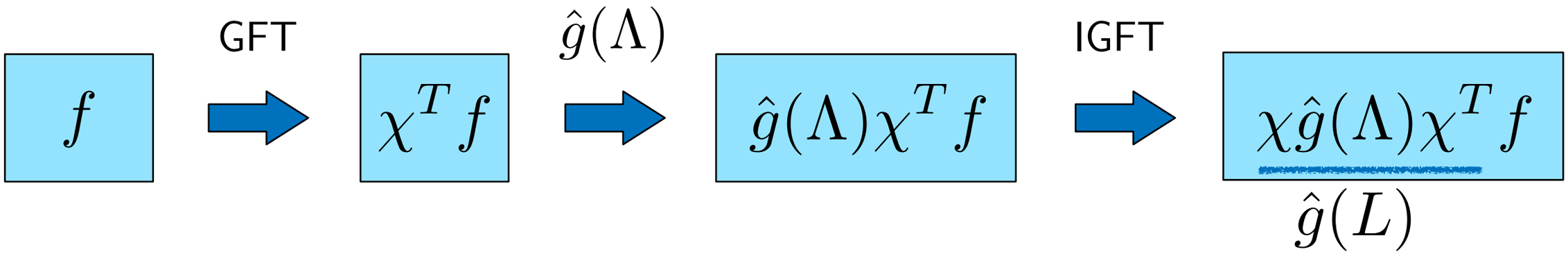
$$y^* = \arg\min_y \{||y - f||_2^2 + \gamma y^T L y\}$$

data fitting term

"smoothness" assumption

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

A practical example



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

$$y^* = \arg\min_y \{||y - f||_2^2 + \gamma y^T L y\}$$

data fitting term

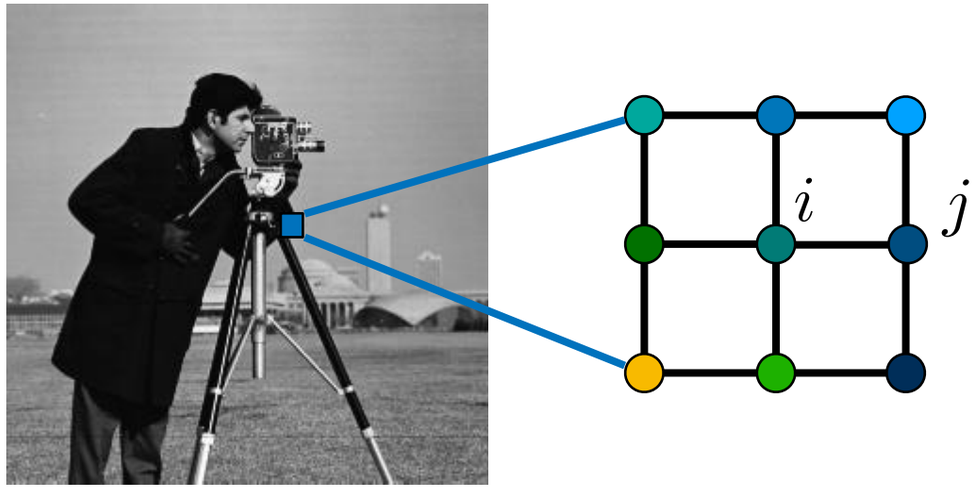
"smoothness" assumption

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

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

A practical example

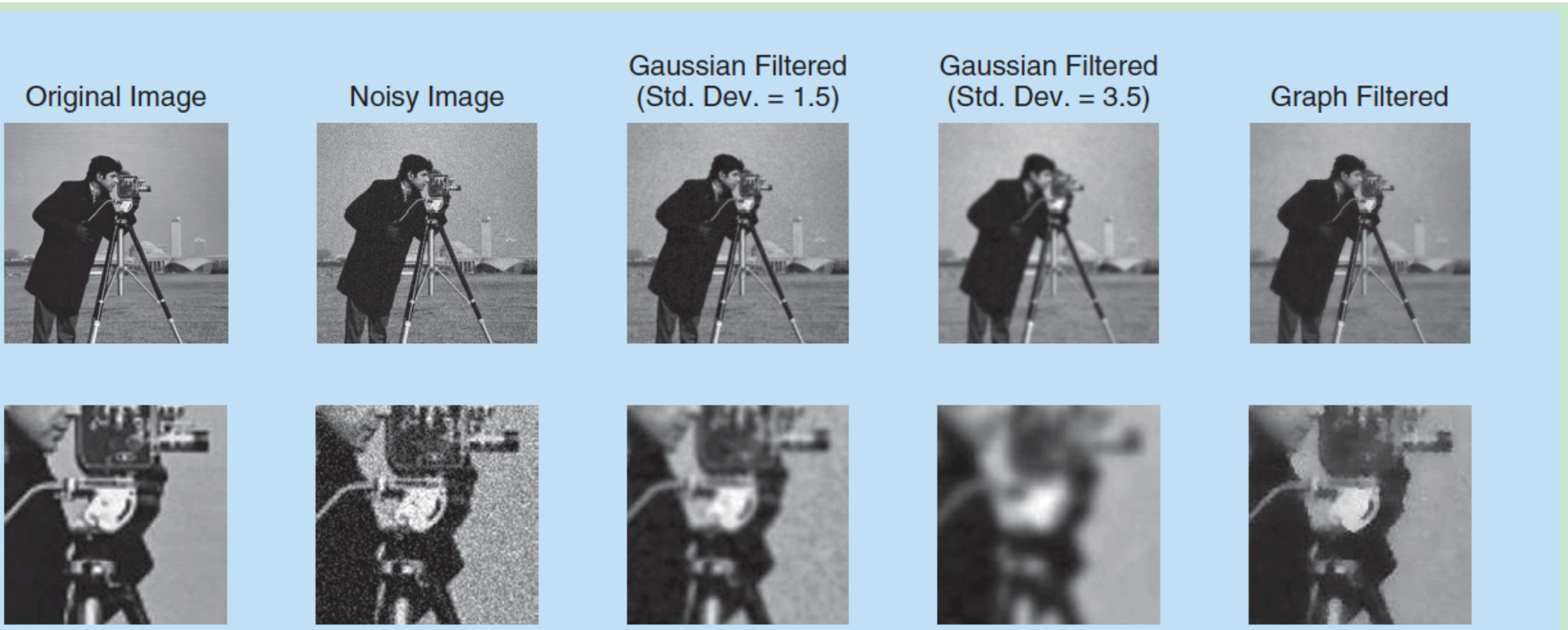
- noisy image as observed noisy graph signal
- regular grid graph (weights inversely proportional to pixel value difference)



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

A practical example

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

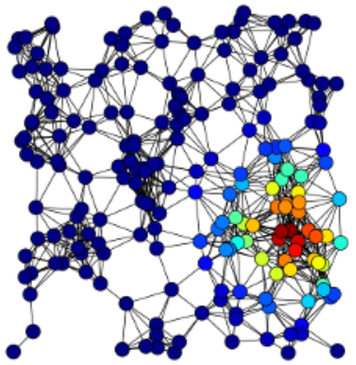


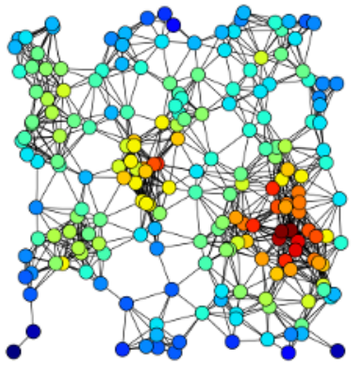
[FIGS2] **Image denoising via classical Gaussian filtering and graph spectral filtering.**

Shuman et al., "The emerging field of signal processing on graphs," IEEE SPM, 2013.

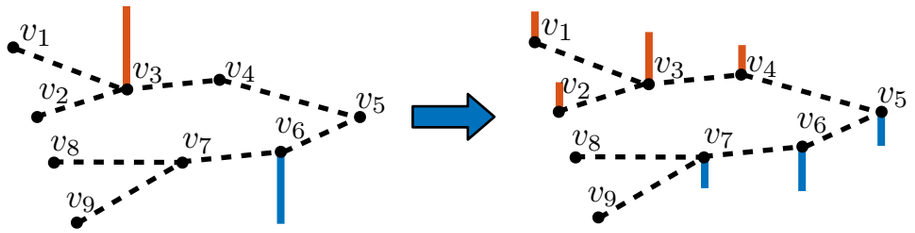
GSP and the literature

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

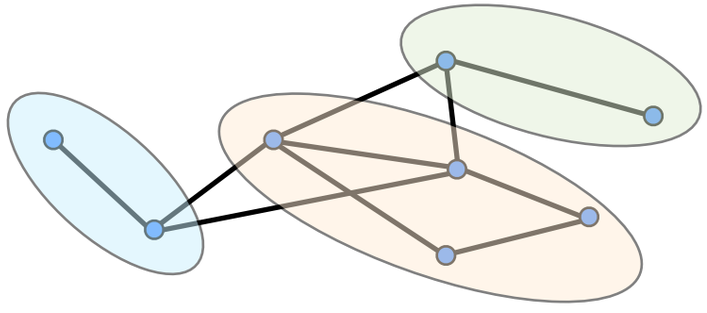




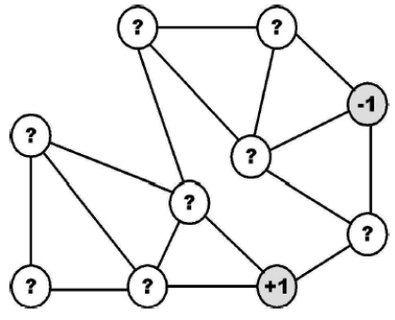
network science (node centrality)



diffusion on graphs



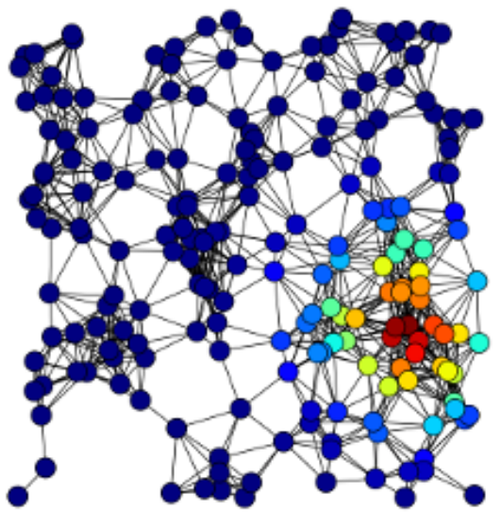
unsupervised learning (dimensionality reduction, clustering)



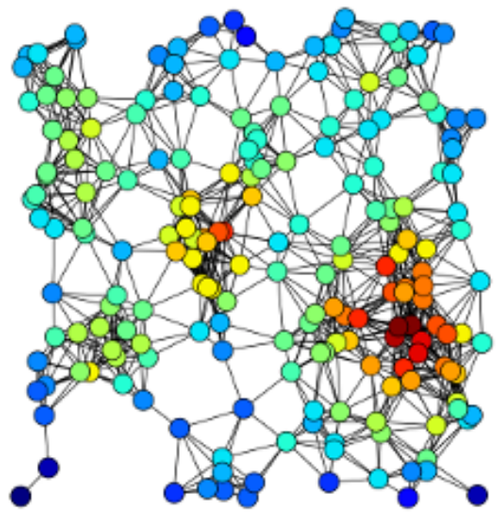
semi-supervised learning

Network centrality

eigenvector centrality



degree centrality



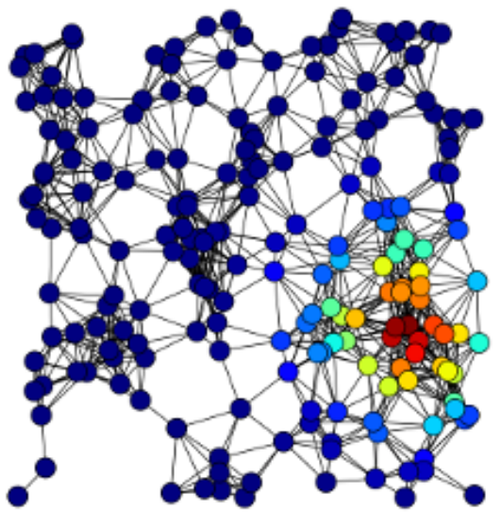
$$Wx = \lambda_{\max} x \qquad\qquad d = [d(v_1), \cdots, d(v_N)]$$

PageRank: http://www.ams.org/publicoutreach/feature-column/fcarc-pagerank

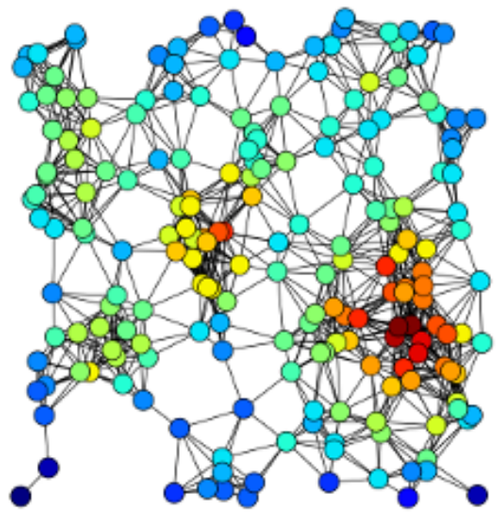
Sandryhaila and Moura, "Discrete signal processing on graphs," IEEE TSP, 2013.

Network centrality

eigenvector centrality



degree centrality



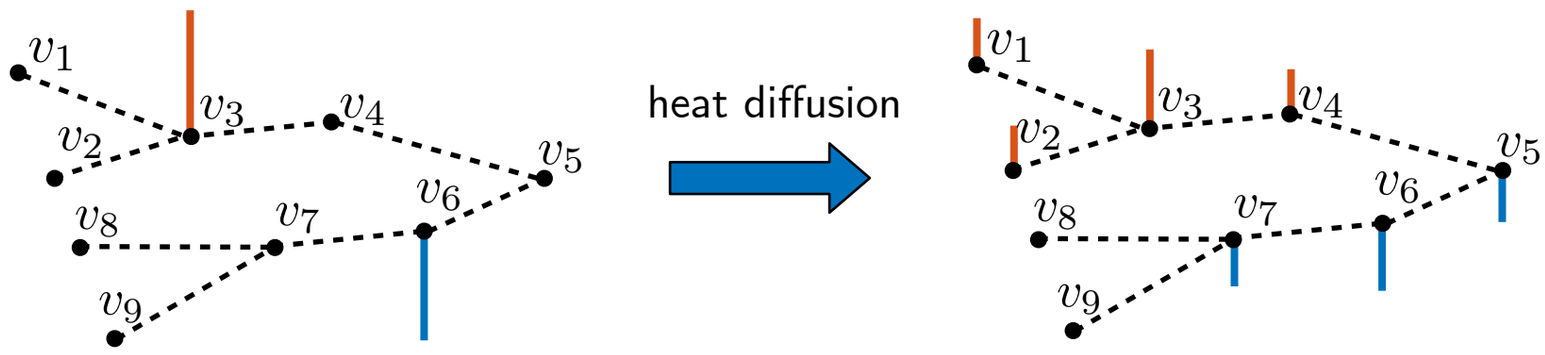
$$Wx = \lambda_{\max} x \qquad d = [d(v_1), \cdots, d(v_N)]$$

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

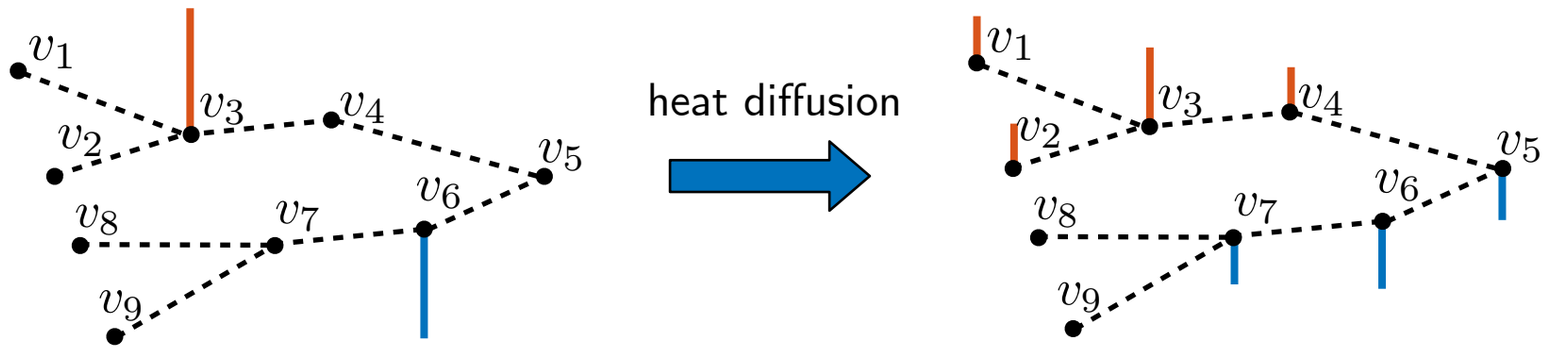
PageRank: http://www.ams.org/publicoutreach/feature-column/fcarc-pagerank

Sandryhaila and Moura, "Discrete signal processing on graphs," IEEE TSP, 2013.

Diffusion on graphs



Diffusion on graphs



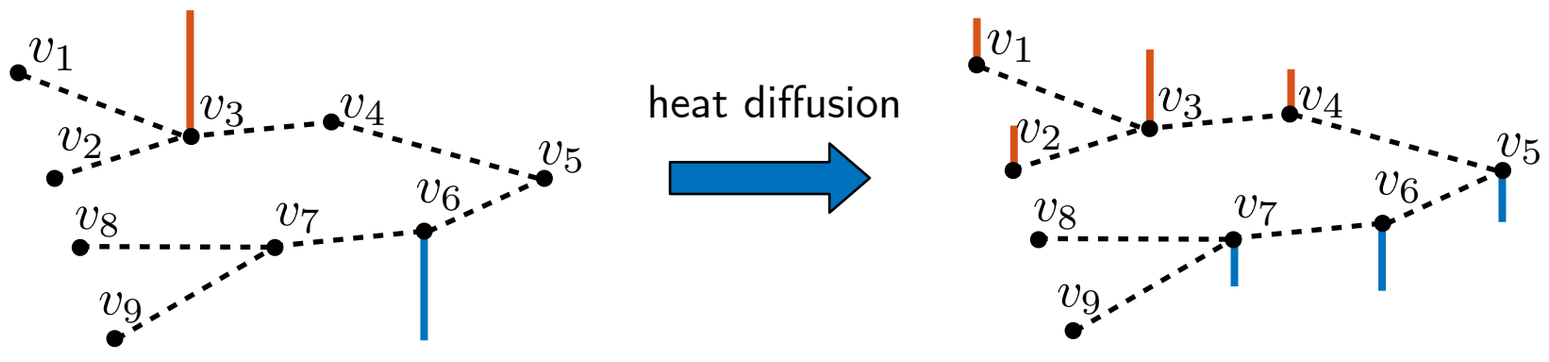
$$\frac{\partial x}{\partial \tau} + Lx = 0$$

$$x(v, 0) = x_0(v)$$

$$\Longrightarrow \quad x(v, \tau) = e^{-\tau L} x_0(v)$$

Smola and Kondor, "Kernels and regularization on graphs," COLT, 2003.

Diffusion on graphs



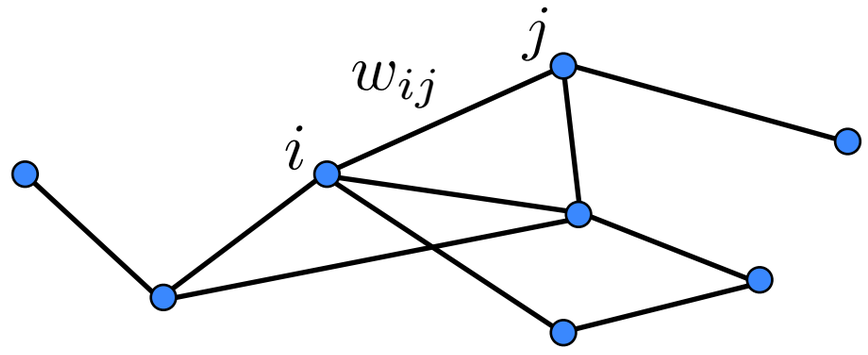
$$\frac{\partial x}{\partial \tau} + Lx = 0$$
$$x(v, 0) = x_0(v)$$

$$\Longrightarrow \quad x(v, \tau) = e^{-\tau L} x_0(v)$$

- 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

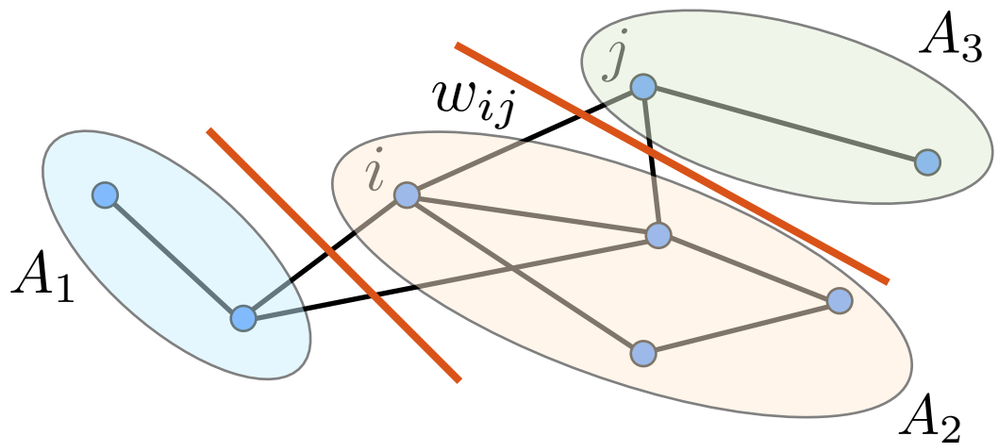
Smola and Kondor, "Kernels and regularization on graphs," COLT, 2003.

Graph clustering (community detection)

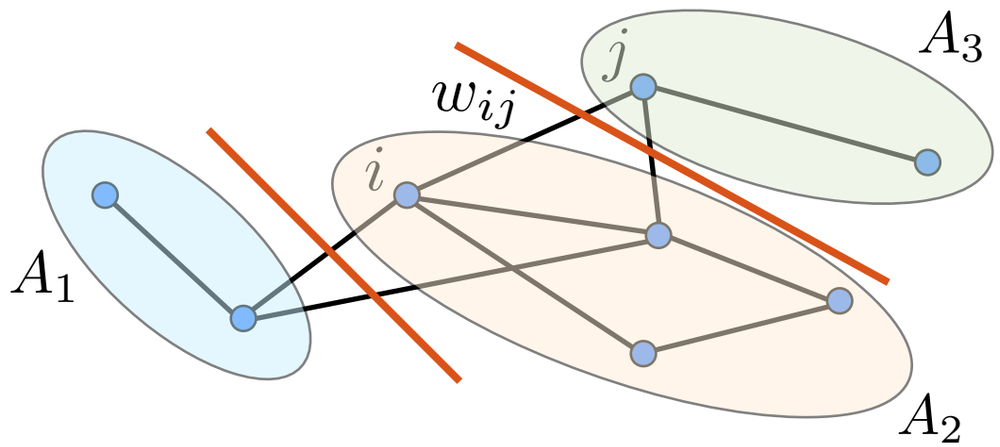


von Luxburg, "A tutorial on spectral clustering," Statistics and Computing, 2007.

Graph clustering (community detection)

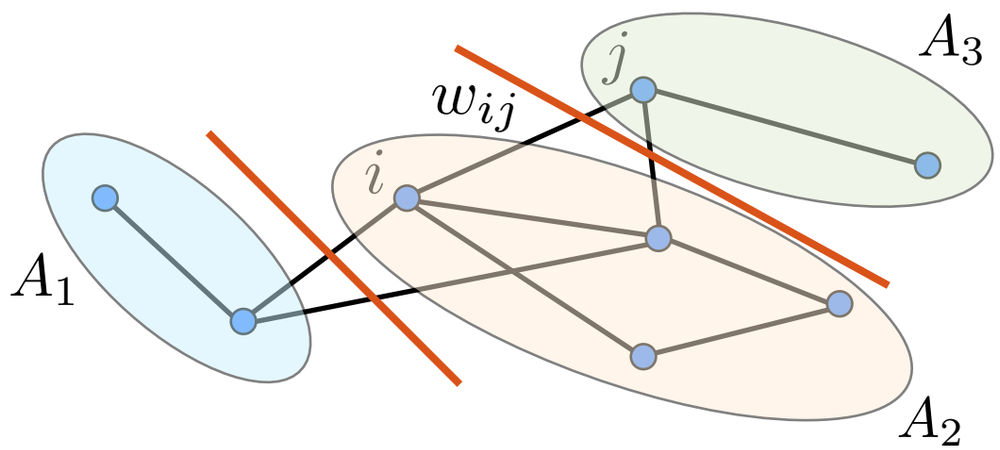


Graph clustering (community detection)



$$NCut(A_1, ..., A_k) = \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{vol(A_i)}$$

Graph clustering (community detection)

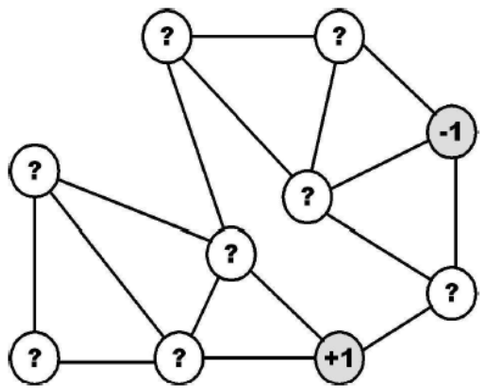


$$NCut(A_1, ..., A_k) = \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{vol(A_i)}$$

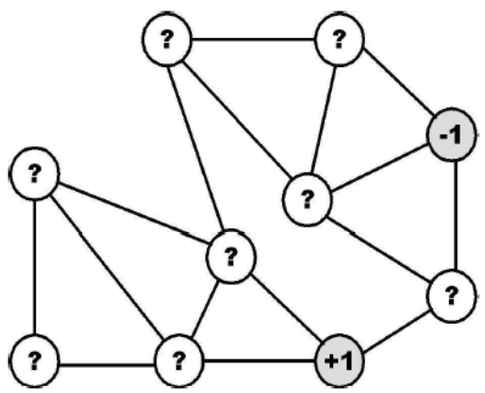
- first k eigenvectors of graph Laplacian provide solution to graph cut minimisation
- eigenvectors of graph Laplacian enable a Fourier-like analysis for graph signals

von Luxburg, "A tutorial on spectral clustering," Statistics and Computing, 2007.

Semi-supervised learning



Semi-supervised learning

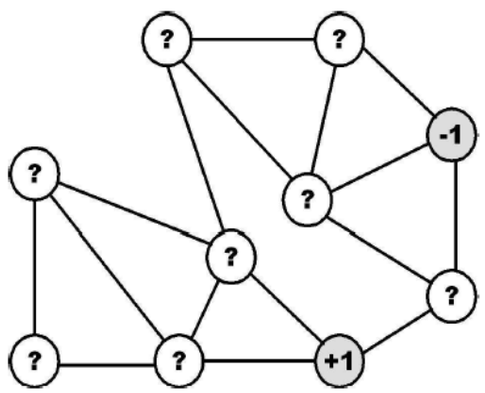


$$y: \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ +1 \\ 0 \\ 0 \\ -1 \\ 0 \end{bmatrix}$$

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

Zhu, "Semi-supervised learning with graphs," Ph.D. dissertation, CMU, 2005.

Semi-supervised learning



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

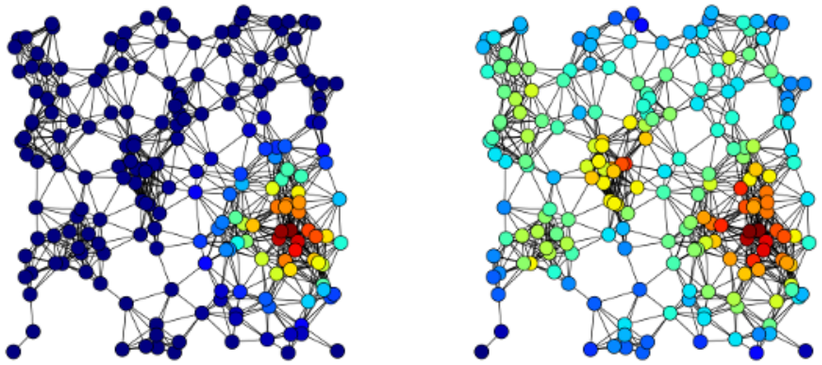
$$\min_{x \in \mathbb{R}^N} ||y - x||_2^2 + \alpha \; x^T L x,$$

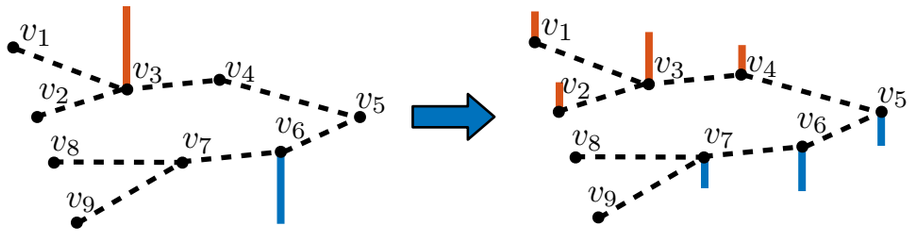
- learning by assuming smoothness of predicted labels (label propagation)
- this is equivalent to a denoising problem for graph signal y

Zhu, "Semi-supervised learning with graphs," Ph.D. dissertation, CMU, 2005.

GSP and the literature

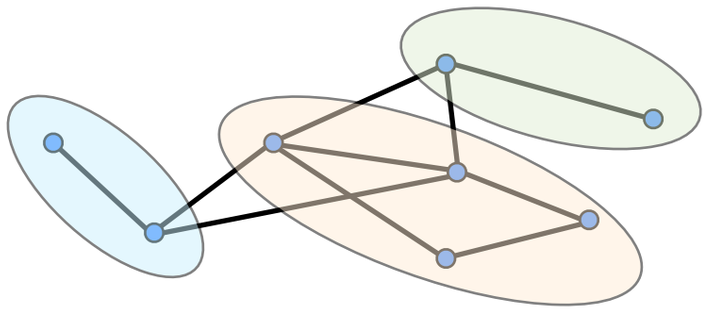
centrality, diffused information, cluster membership, node labels (and node features in general) can ALL be viewed as graph signals

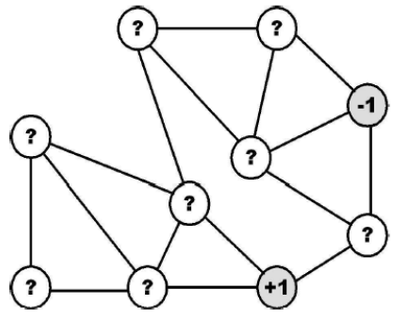




network science

diffusion on graphs





unsupervised learning (dimensionality reduction, clustering)

semi-supervised learning