From Graphs to Graph Structured Data

Dr Dorina Thanou

April 24, 2023

Recap from previous class

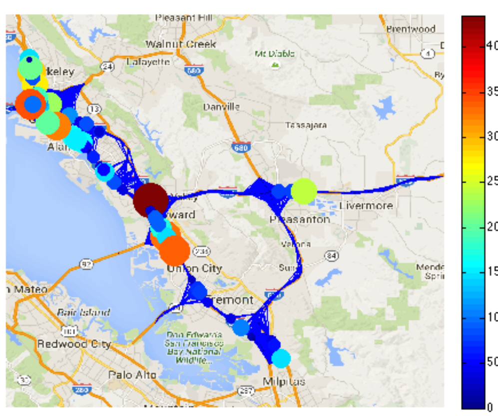
- Networks/graphs are either indicated by the application or constructed from the data

- Spectral graph theory reveals significant properties of the network
  - Spectrum tells us a lot about connectivity, bottlenecks, diameter
  - Eigenvalues provide a notion of frequency
  - Eigenvectors are smooth functions on the graph

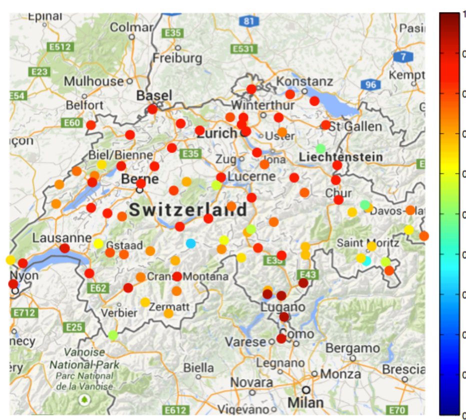
- It has applications in network tasks, where preserving geometry is crucial

Going beyond graph structure

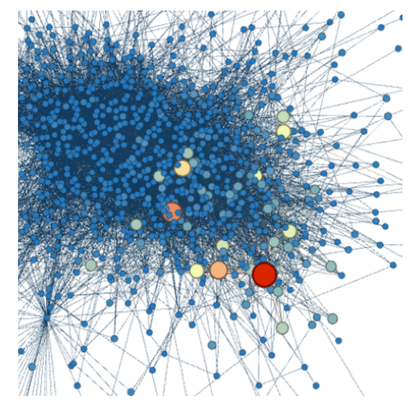
- Very often data comes with additional features
  - Not only graphs, but attributes on the nodes of the graph
- Key question: What is the interplay between graph structure and node attributes?



Transportation networks

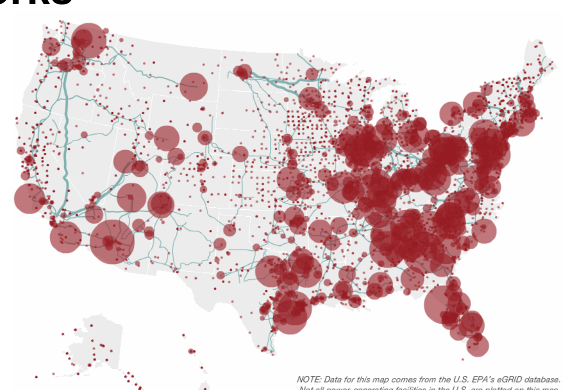


Weather networks

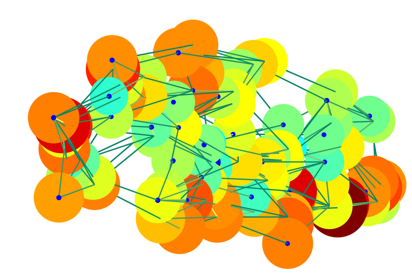


Social networks

Disease spreading networks



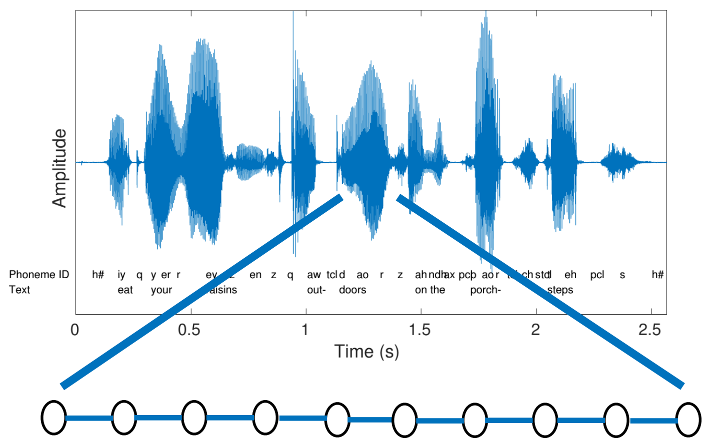
Electric grid networks



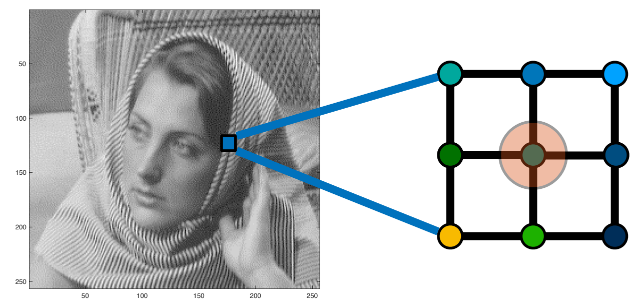
Biological networks

Graph structured data

- In classical applications, data often lives on a regular domain

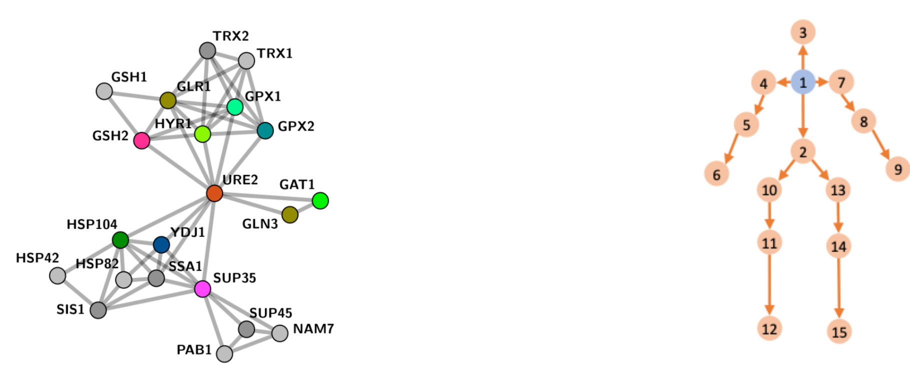


Domain: line



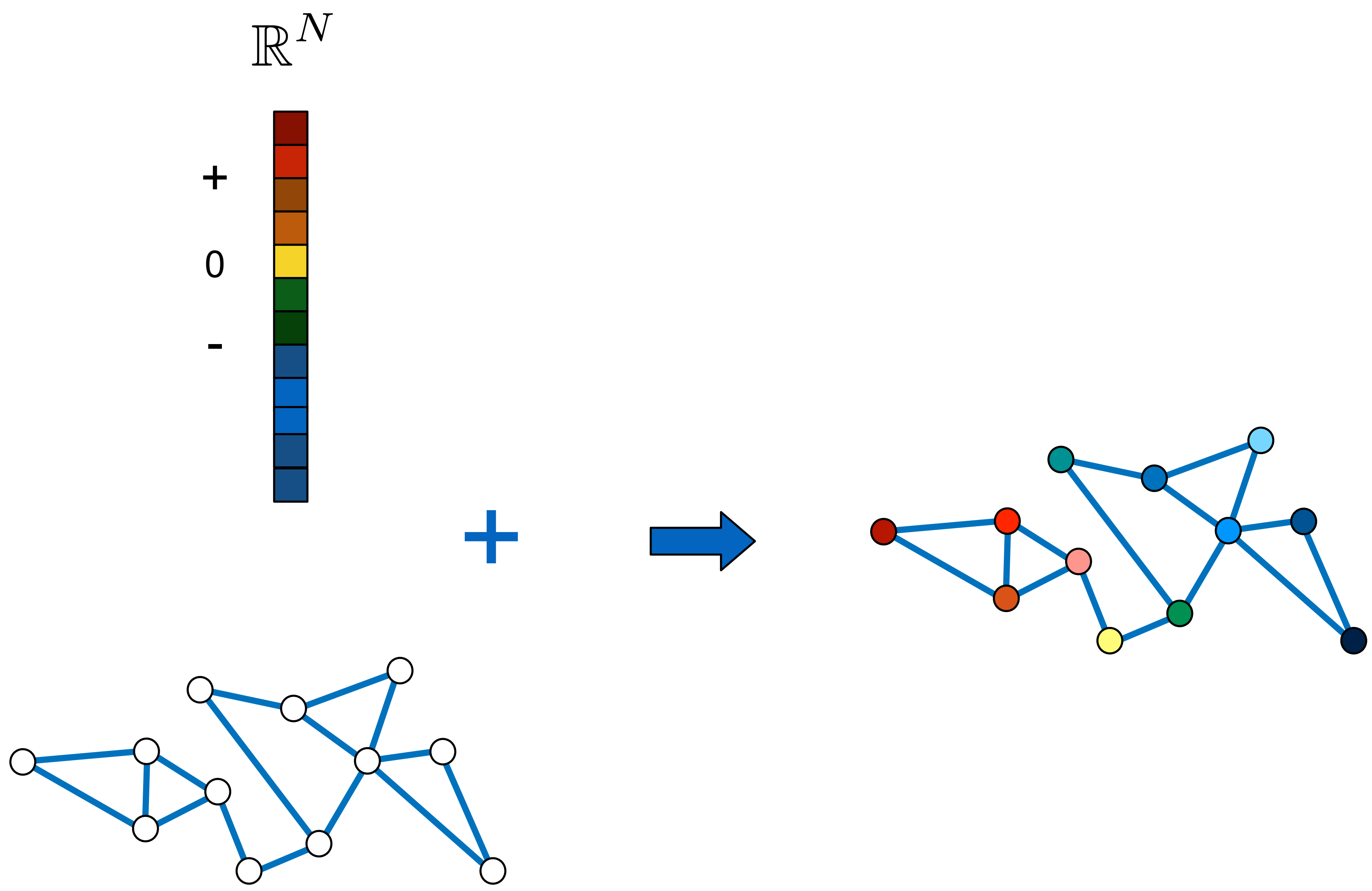
Domain: grid

- Weighted graphs capture the geometric structure of complex, i.e., irregular, domains



Domain: irregular graph

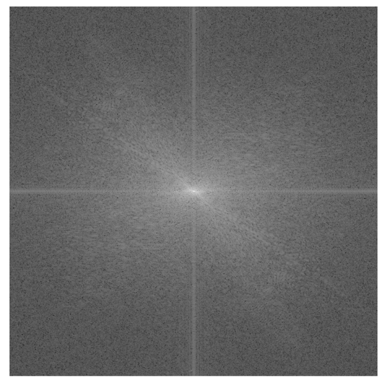
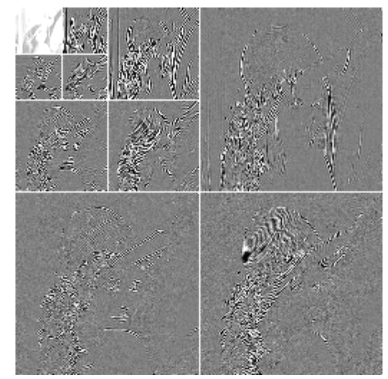
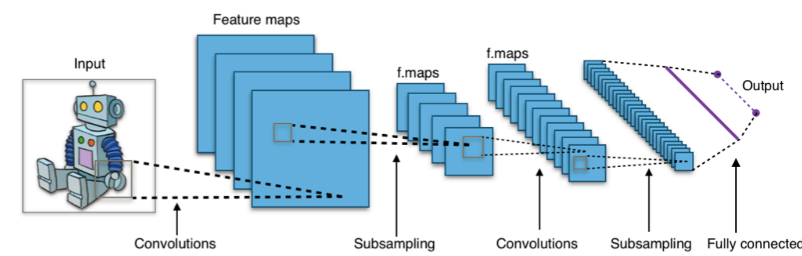
Processing graph structured data



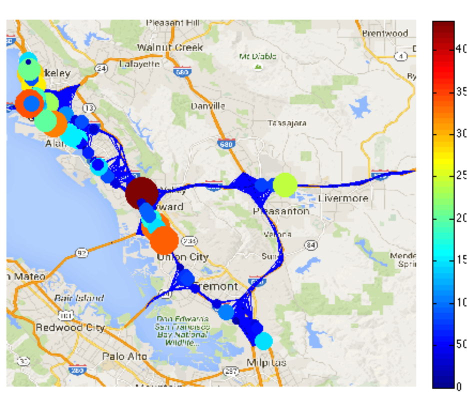
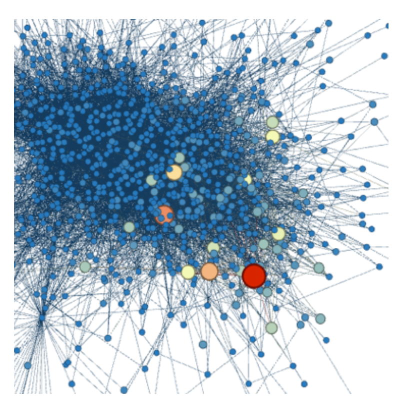
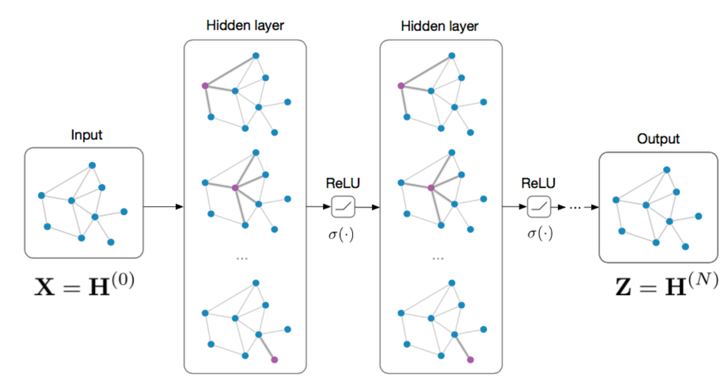
How can we extract useful information by taking into account both **structure (edges)** and **data (values/features on vertices)**?

Representation of structured data

- Traditional approaches: Harmonic analysis on Euclidean domain (e.g., Fourier, wavelets), (deep) representation learning

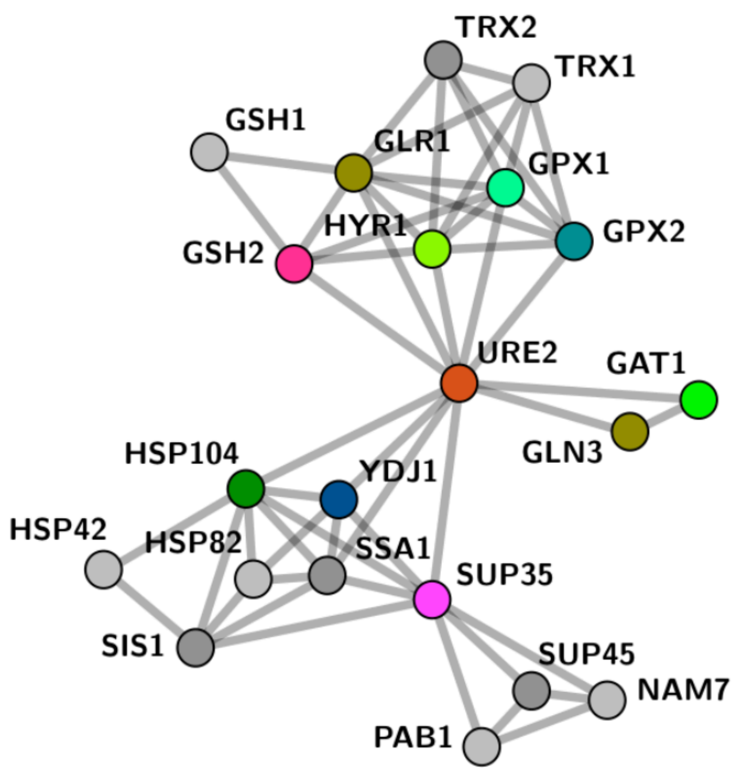





- Irregular structures: how do we generalize such notions to graph settings?

In this lecture…

- How can we extract information from graph structured data, using well-defined notions from signal processing?



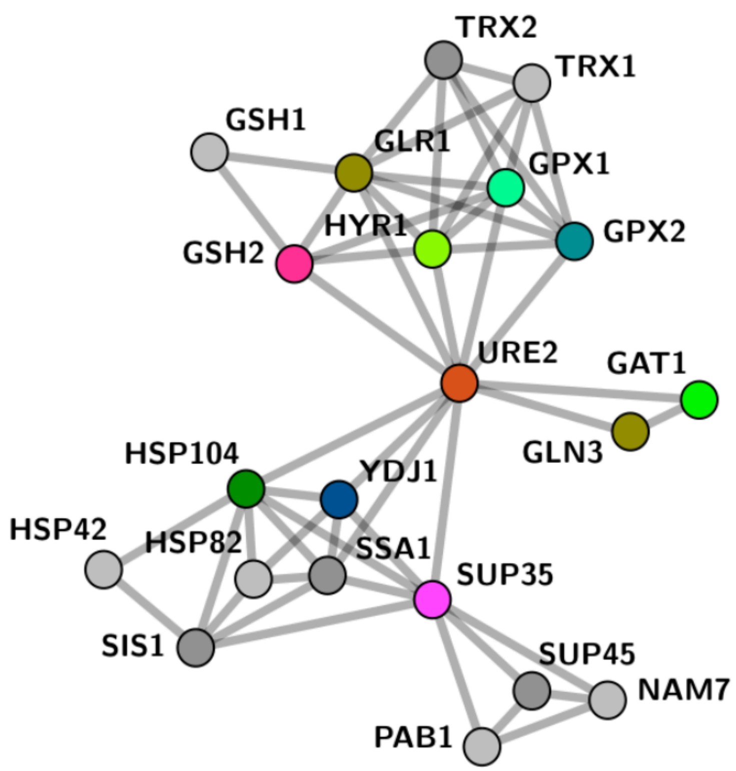
Graph representations → **Actionable knowledge**

e.g., node/graph classification, signal inpainting/denoising

$$X, \mathcal{G} \rightarrow f(X, \mathcal{G}) \rightarrow Y$$

In this lecture…

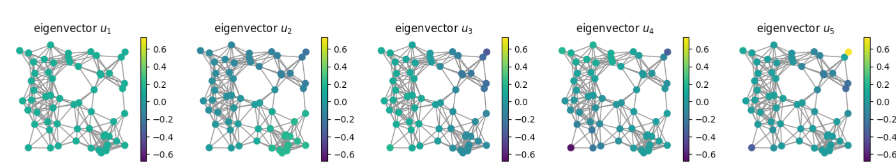
- How can we extract information from graph structured data, using well-defined notions from signal processing?



Graph representations

Graph Signal Processing (GSP)

Actionable knowledge



e.g., node/graph classification, signal inpainting/denoising

$$X, \mathcal{G} \quad \rightarrow \quad f(X, \mathcal{G}) \quad \rightarrow \quad Y$$

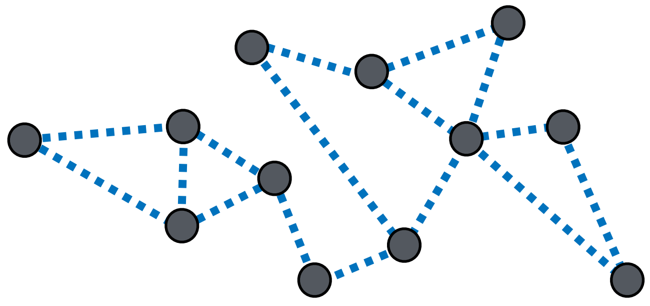
Outline

- Graphs and signals on graphs
- Graph Fourier transform
- Filtering on graphs
- Spectral graph convolution
- Applications
  - Regularization on graphs
  - Compression
  - Knowledge discovery

Graphs and signals on graphs

- **Irregular domain:** connected, undirected, weighted graph of N nodes

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$$



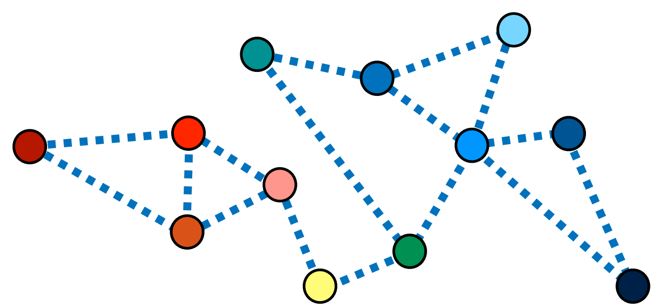
- Neighborhood of node i:

$$\mathcal{N}_i = \{j : (i,j) \in \mathcal{E}\}$$

- **Graph description:**
  - Degree matrix D: diagonal matrix with sum of weights of incident edges
  - Laplacian matrix L: $L = D - W, \quad L = \chi \Lambda \chi^T$
    - Complete set of orthonormal eigenvectors $\chi = [\chi_1, \chi_2, ..., \chi_N]$
    - Real, non-negative eigenvalues $0 = \lambda_1 < \lambda_2 \leq \lambda_3 \leq ... \leq \lambda_N$

Signal on the graph or graph signal

- A function $f : \mathcal{V} \rightarrow \mathbb{R}^N$ that assigns real values to each vertex of the graph
- It is defined on the vertices of the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$

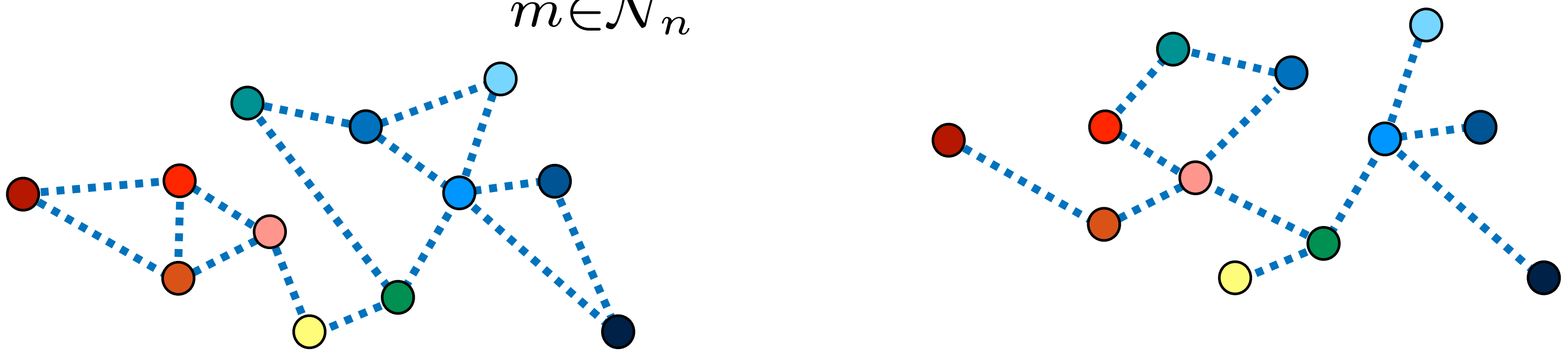


- Often represented as a vector $f \in \mathbb{R}^N$, where $f(i)$ is the signal value at node i
- The ordering of the vector follows the ordering of the adjacency matrix

Graph Laplacian operator

- Combinatorial Laplacian: differential operator that computes the pairwise difference between signal values in the neighborhood

$$\big(Lf\big)(n) = \sum_{m \in \mathcal{N}_n} W_{n,m}[f(n) - f(m)]$$



$f^T L_1 f = 0.15$

Smooth signal

$f^T L_2 f = 1.8$

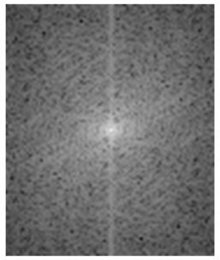
Non-smooth signal

- It is helpful for defining global smoothness on the graph:

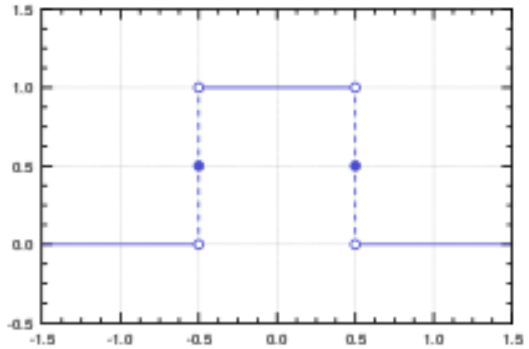
$$f^T L f = \sum_{n \in \mathcal{V}} \sum_{m \in \mathcal{N}_n} W_{n,m}[f(n) - f(m)]^2$$

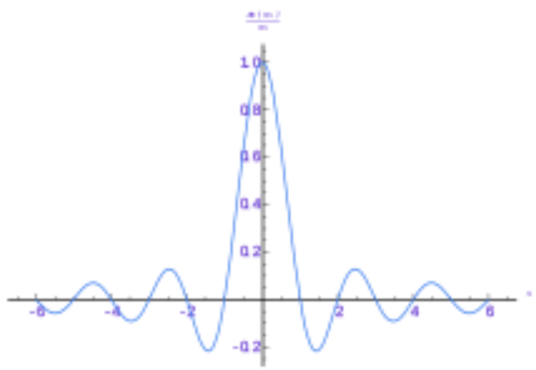
The Fourier transform

- One of the most fundamental notions in signal processing/ analysis



- A mathematical transform that decomposes functions depending on space or time into functions depending on spatial or temporal frequency



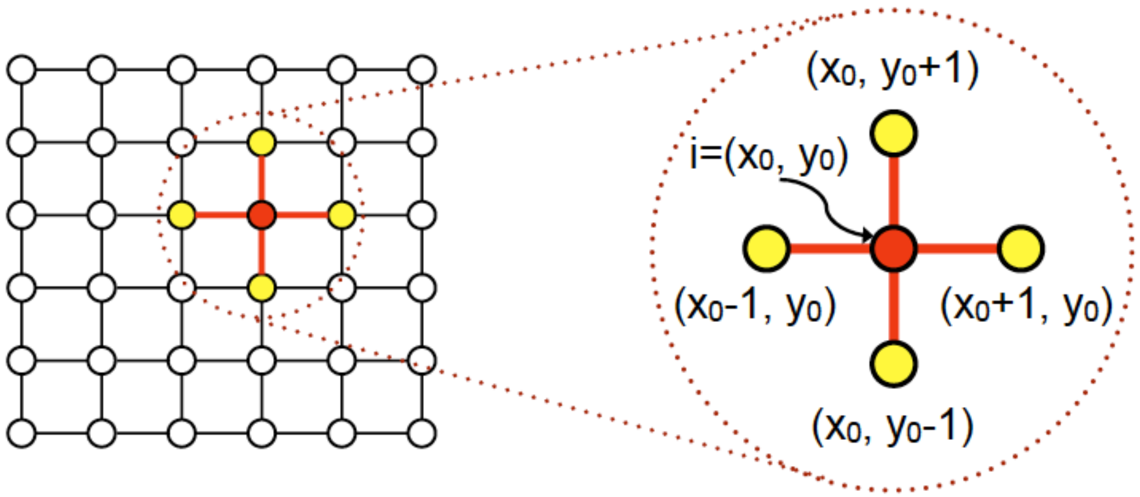


How can we define the Graph Fourier transform for graph structured data?

Recall that …

- The Laplacian matrix is the graph analogue to the Laplace operator on continuous functions!

- Example from previous lecture: Unweighted grid graph



$$
\begin{aligned}
-Lf(i) = \ & [f(x_0+1, y_0) - f(x_0, y_0)] - [f(x_0, y_0) - f(x_0-1, y_0)] \\
& + [f(x_0, y_0+1) - f(x_0, y_0)] - [f(x_0, y_0) - f(x_0, y_0-1)]
\end{aligned}
$$

$$
\sim \frac{\partial^2 f}{\partial x^2}(x_0, y_0) + \frac{\partial^2 f}{\partial y^2}(x_0, y_0) = (\Delta f)(x_0, y_0)
$$

A notion of frequency on the graph

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

one-dimensional Laplace operator: $\dfrac{d^2}{dx^2}$

$\Downarrow$

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

$\Downarrow$

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

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

graph Laplacian: L

$\Downarrow$

eigenvectors: χ_ℓ

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

$\Downarrow$

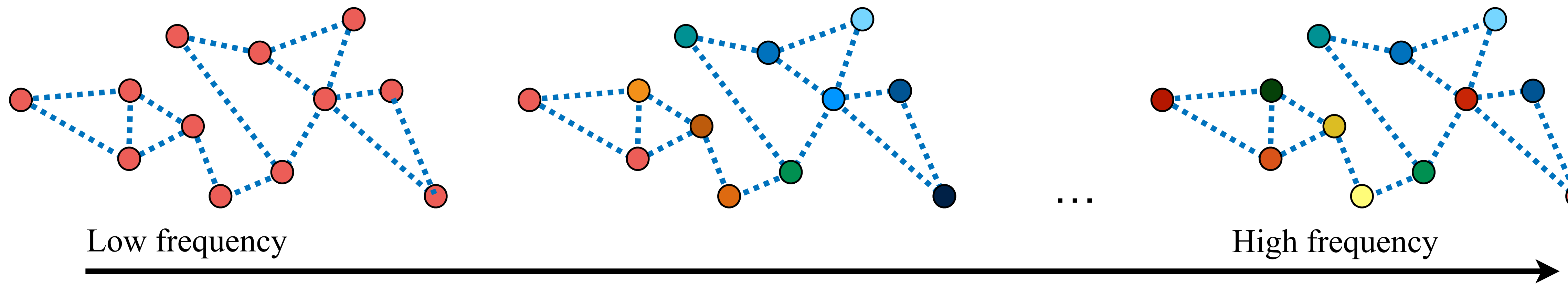
Graph FT: $\hat{f}(\ell) = \langle \chi_\ell, f \rangle = \boxed{\sum_{i=1}^{N}} \boxed{\chi_\ell^*(i)} \boxed{f(i)}$

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

FT: Fourier Transform

Graph Fourier transform

- The eigenvectors of the Laplacian provide a harmonic analysis of graph signals



$\chi_1^T L \chi_1 = \lambda_1$ $\chi_2^T L \chi_2 = \lambda_2$ $\chi_N^T L \chi_N = \lambda_N$

Graph Fourier Transform: $$\hat{f}(\lambda_\ell) = < f, \chi_\ell > = \sum_{n=1}^{N} f(n) \chi_\ell^T(n), \quad \ell = 1, 2, ..., N$$

- By exploiting the orthonormality of the eigenvectors, we obtain:

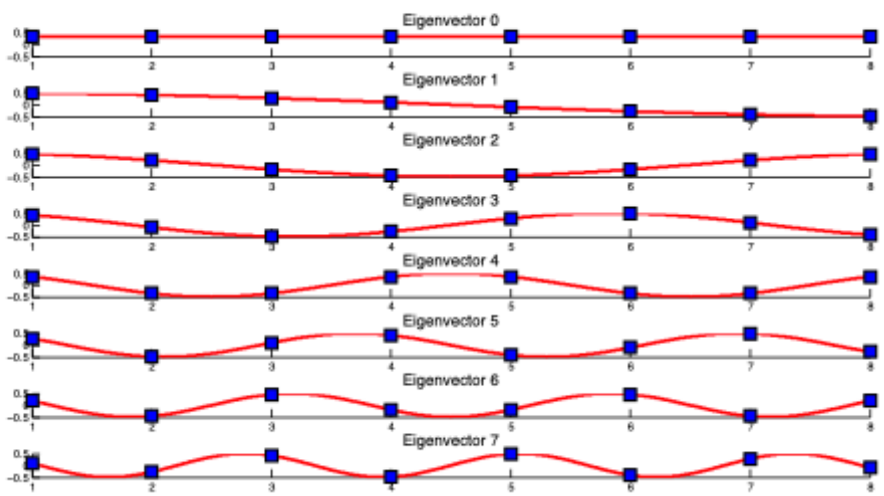
Inverse Graph Fourier Transform: $$f(n) = \sum_{\ell=1}^{N} \hat{f}(\lambda_\ell) \chi_\ell(n), \quad \forall n \in \mathcal{V}$$

A special case: The path graph

- The eigenvalues of the graph Laplacian of the unweighted path graph are given by

$$\lambda_\ell = 2 - 2\cos(\frac{\pi \ell}{N}), \quad \forall \ell \in 1, 2, ..., N$$

- The corresponding eigenvectors are

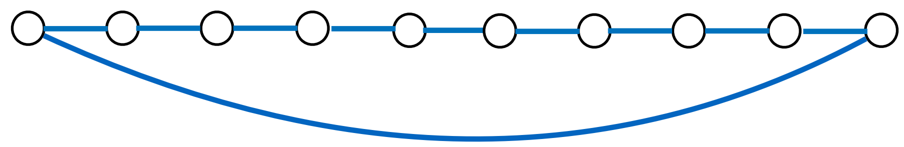


$$\chi_1(n) = \frac{1}{\sqrt{N}}, \quad \forall \quad n = 1, 2, ..., N$$

$$\chi_\ell(n) = \sqrt{\frac{2}{N}}\cos(\frac{\pi \ell (n - 0.5)}{N}), \quad \forall \quad \ell = 2, 3, ..., N$$

Basis vectors of Discrete Cosine Transform used in JPEG for example

A special case: The ring graph



- The eigenvalues of the graph Laplacian of the unweighted ring graph are given by

$$\lambda_\ell = 2 - 2\cos(\frac{2\pi\ell}{N}), \quad \forall \ell \in 1, 2, ..., N$$

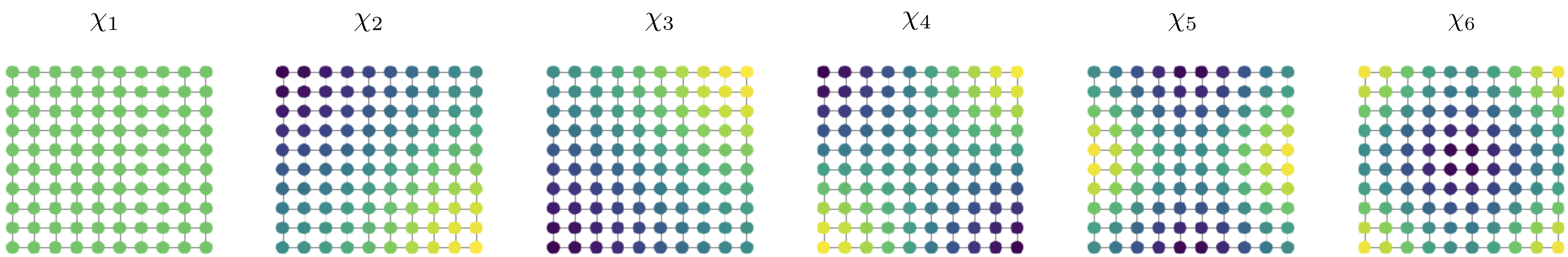
- Since the Laplacian is a circulant matrix, the corresponding eigenvectors are

$$\chi_\ell = \frac{1}{\sqrt{N}}\left[1, \omega^{\ell}, \omega^{2\ell}, ..., \omega^{(N-1)\ell}\right]^T, \quad \omega = e^{\frac{2\pi j}{N}}$$

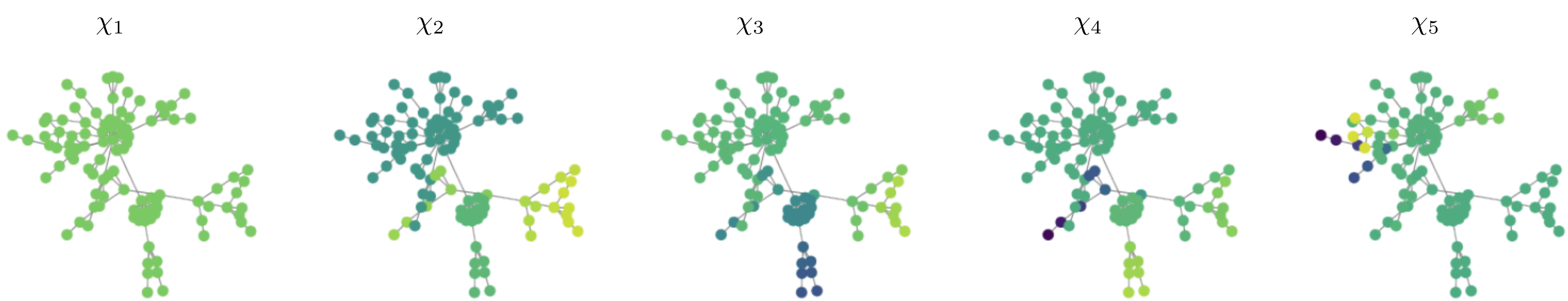
Discrete Fourier Transform (DFT)

Some other examples

- The regular grid graph

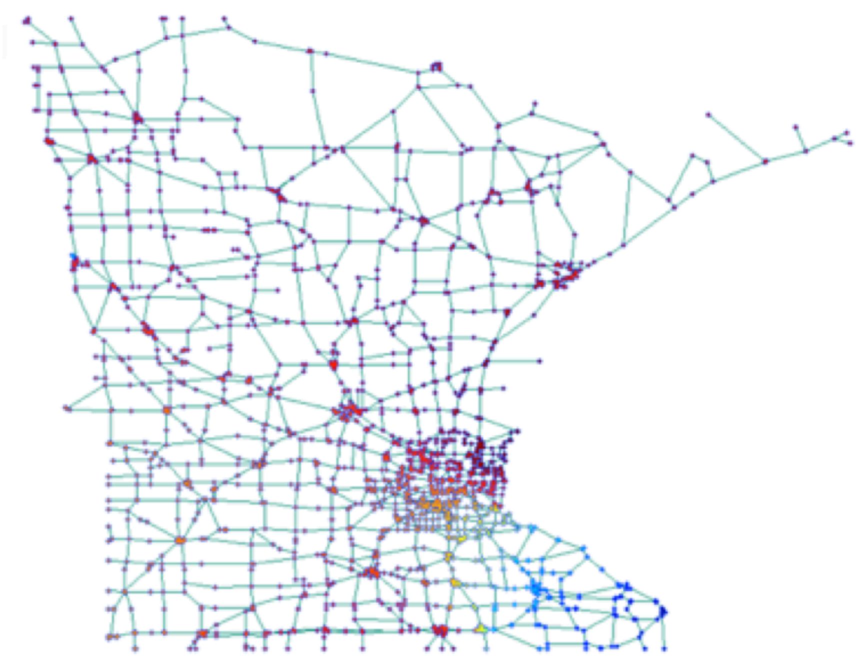


- Barabasi-Albert scale-free network

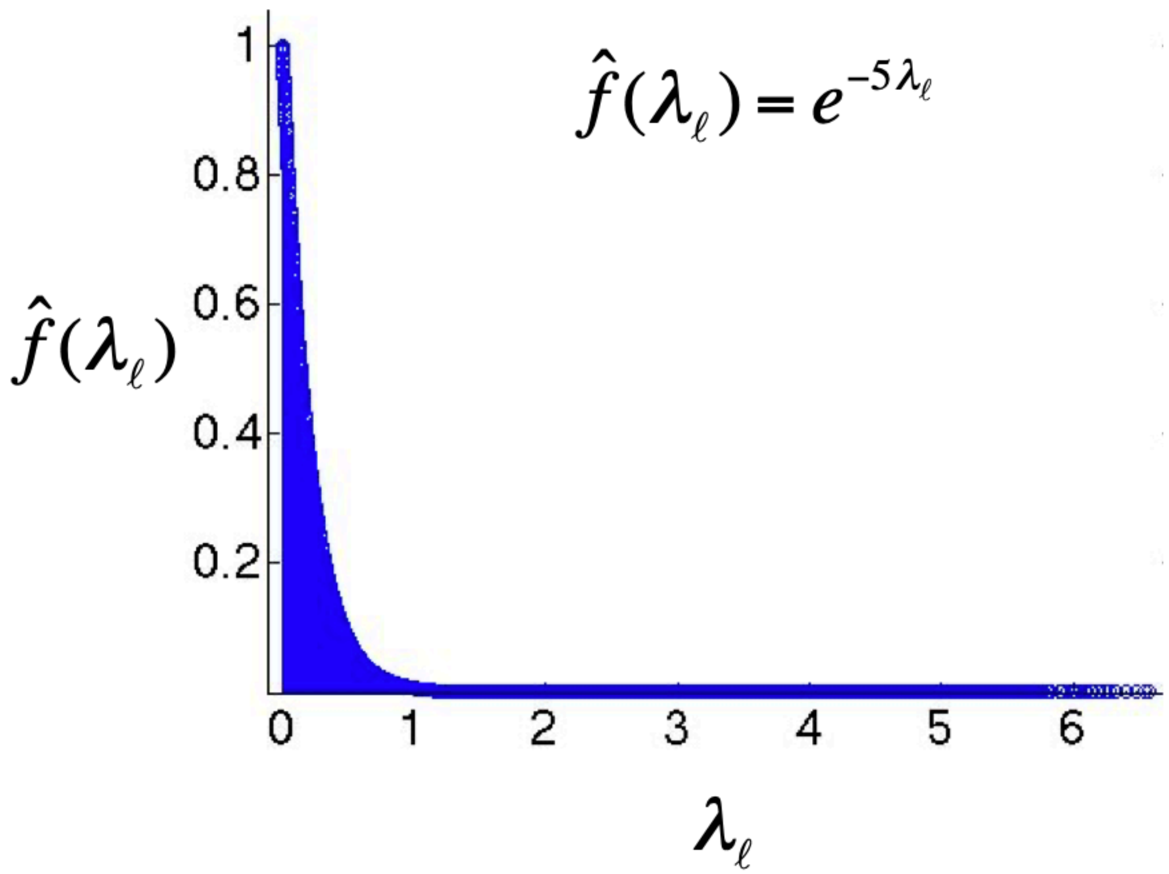


A dual representation of the graph signal

Reminder:
$$\hat{f}(\lambda_\ell) = < f, \chi_\ell > = \sum_{n=1}^{N} f(n)\chi_\ell^T(n), \quad \ell = 1, 2, ..., N$$
$$f(n) = \sum_{\ell=1}^{N} \hat{f}(\lambda_\ell)\chi_\ell(n), \quad \forall n \in \mathcal{V}$$

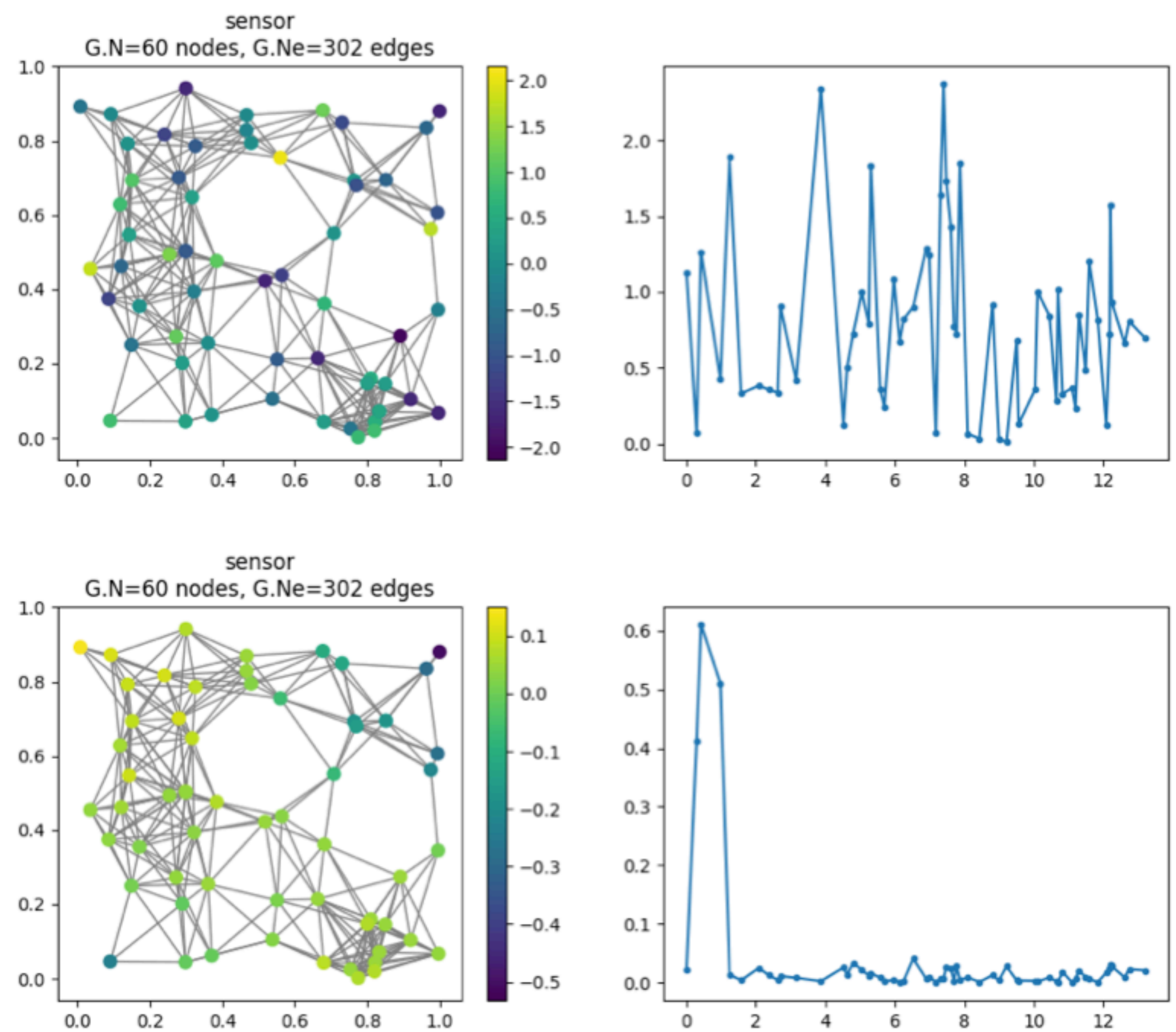


Vertex domain



Graph spectral domain

Dual representation continued



Vertex domain **Graph spectral domain**

- The spectral domain representation tells us a lot about the variation of the signal in the vertex domain

Classical frequency filtering

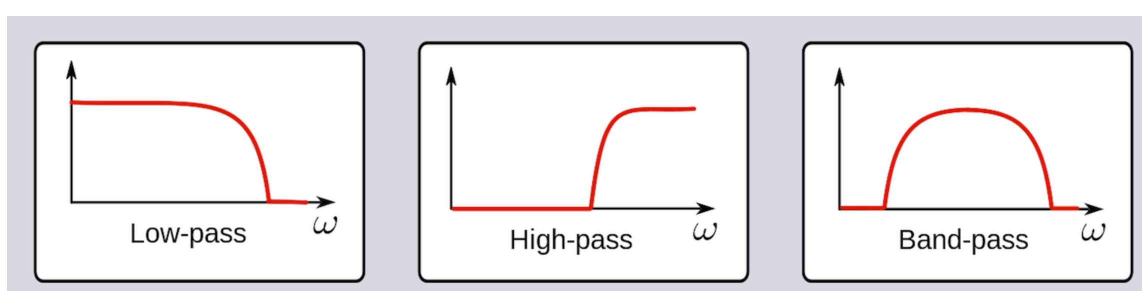
- It is given by amplifying or attenuating the contributions of some Fourier bases
  - The FT is defined as $\widehat{f}(\omega) = \int (e^{j\omega x})^* f(x)dx, \quad f(x) = \int \widehat{f}(\omega)e^{j\omega x}d\omega$
  - Filtering a signal f with a transfer function $\hat{g}(\cdot)$ is defined as follows

$$\boxed{f} \xrightarrow{\text{FT}} \boxed{\widehat{f}(\omega)} \xrightarrow{\hat{g}(\omega)} \boxed{\widehat{g}(\omega)\widehat{f}(\omega)} \xrightarrow{\text{IFT}} \boxed{g * f}$$

Classical frequency filtering

- It is given by amplifying or attenuating the contributions of some Fourier bases
  - The FT is defined as $\widehat{f}(\omega) = \int (e^{j\omega x})^* f(x) dx, \quad f(x) = \int \widehat{f}(\omega) e^{j\omega x} d\omega$
  - Filtering a signal f with a transfer function $\hat{g}(\cdot)$ is defined as follows

$$f \xrightarrow{\text{FT}} \widehat{f}(\omega) \xrightarrow{\hat{g}(\omega)} \widehat{g}(\omega)\widehat{f}(\omega) \xrightarrow{\text{IFT}} g * f$$



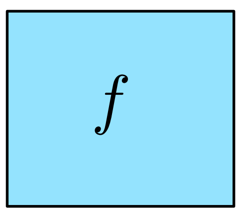
Graph spectral filtering

- It is defined in direct analogy with classical filtering in the frequency domain
  - Filtering a graph signal f with a spectral filter $\hat{g}(\cdot)$ is performed in the graph Fourier domain

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

Graph spectral filtering

- It is defined in direct analogy with classical filtering in the frequency domain
  - Filtering a graph signal f with a spectral filter $\hat{g}(\cdot)$ is performed in the graph Fourier domain



Shuman et al., “The emerging field of signal processing on graphs”, IEEE Signal Process. Mag., 2013

Graph spectral filtering

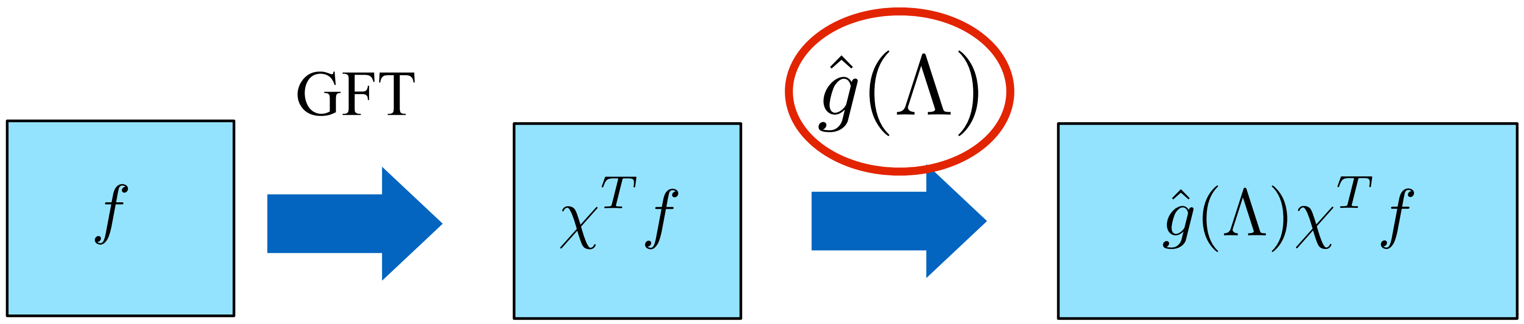
- It is defined in direct analogy with classical filtering in the frequency domain
  - Filtering a graph signal f with a spectral filter $\hat{g}(\cdot)$ is performed in the graph Fourier domain

$$\boxed{f} \xrightarrow{\text{GFT}} \boxed{\chi^T f}$$

Shuman et al., “The emerging field of signal processing on graphs”, IEEE Signal Process. Mag., 2013

Graph spectral filtering

- It is defined in direct analogy with classical filtering in the frequency domain
  - Filtering a graph signal f with a spectral filter $\hat{g}(\cdot)$ is performed in the graph Fourier domain



Shuman et al., “The emerging field of signal processing on graphs”, IEEE Signal Process. Mag., 2013

Graph spectral filtering

- It is defined in direct analogy with classical filtering in the frequency domain
  - Filtering a graph signal f with a spectral filter $\hat{g}(\cdot)$ is performed in the graph Fourier domain

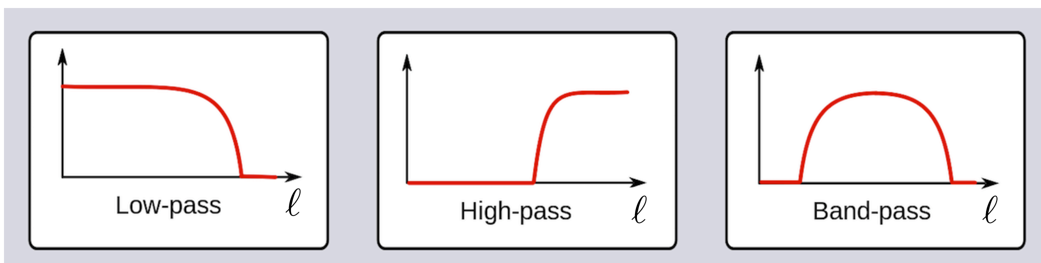
$$f \xrightarrow{\text{GFT}} \chi^T f \xrightarrow{\hat{g}(\Lambda)} \hat{g}(\Lambda)\chi^T f \xrightarrow{\text{IGFT}} \chi\hat{g}(\Lambda)\chi^T f$$

Shuman et al., “The emerging field of signal processing on graphs”, IEEE Signal Process. Mag., 2013

Graph spectral filtering

- It is defined in direct analogy with classical filtering in the frequency domain
  - Filtering a graph signal f with a spectral filter $\hat{g}(\cdot)$ is performed in the graph Fourier domain

$$\boxed{f} \xrightarrow{\text{GFT}} \boxed{\chi^T f} \xrightarrow{\hat{g}(\Lambda)} \boxed{\hat{g}(\Lambda)\chi^T f} \xrightarrow{\text{IGFT}} \boxed{\chi\hat{g}(\Lambda)\chi^T f}$$

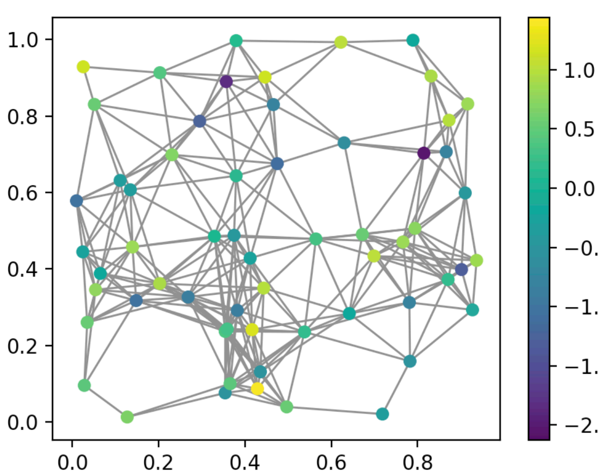


Shuman et al., “The emerging field of signal processing on graphs”, IEEE Signal Process. Mag., 2013

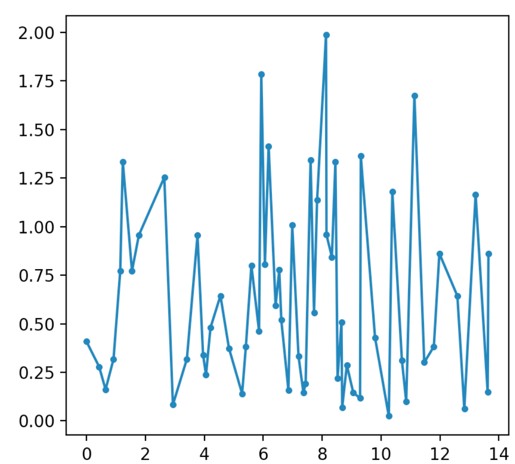
Illustrative example

- Apply a low filter to a graph signal
  - Keep only the first GFT coefficients

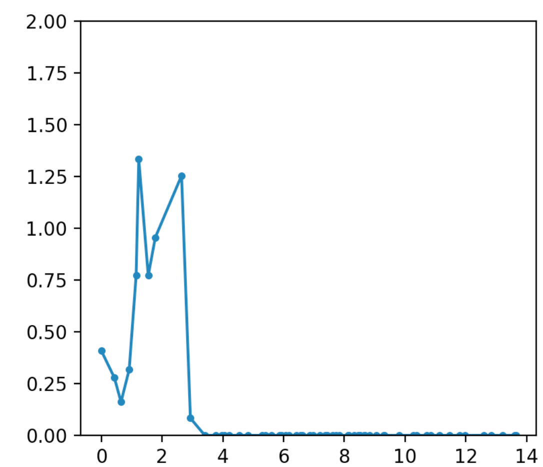
- Recover the filtered signal in the vertex domain
  - The filtered signal is smoother on the graph



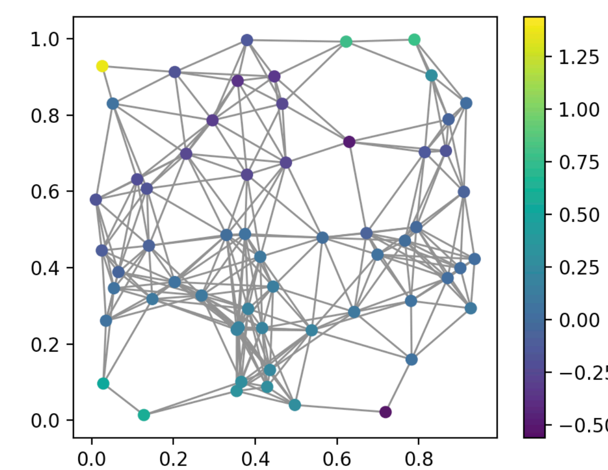
f



$\chi^T f$



$\hat{g}(\Lambda)\chi^T f$

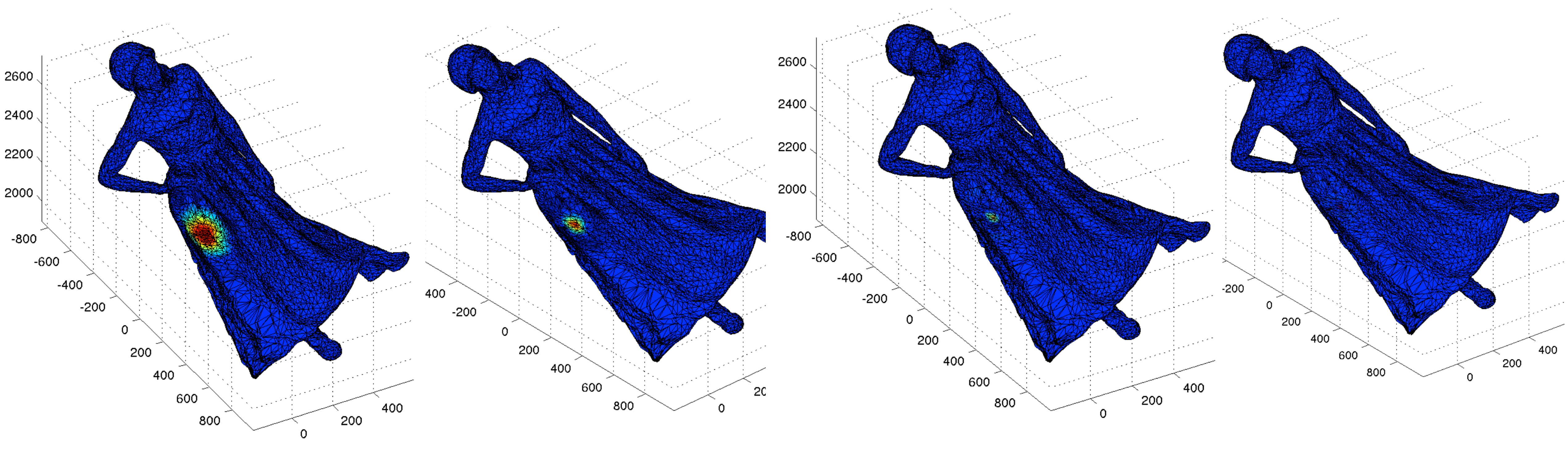


$\tilde{f} = \chi\hat{g}(\Lambda)\chi^T f$

Other graph transforms

- Other graph transforms and dictionaries can be designed by filtering the eigenvalues of the graph Laplacian

- Example: By defining shifted and dilated bandpass filters, we obtain a generalisation of wavelets on the graph



Hammond et al., “Wavelets on graphs via spectral graph theory”, ACHA, 2009

Convolution on graphs

- A mathematical operator that computes the “amount of overlap" between two functions

- Convolution in the time domain is equivalent to multiplication in the frequency domain

	Classical convolution	Convolution on graphs
Time domain	$(f * g)(t) = \int_{-\infty}^{\infty} f(t-\tau) g(\tau) d\tau$	

- More in the following lectures

Convolution on graphs

- A mathematical operator that computes the “amount of overlap" between two functions

- Convolution in the time domain is equivalent to multiplication in the frequency domain

	Classical convolution	Convolution on graphs
Time domain	$(f * g)(t) = \int_{-\infty}^{\infty} f(t-\tau)g(\tau)d\tau$	
Frequency domain	$\widehat{(f * g)}(\omega) = \hat{f}(\omega) \cdot \hat{g}(\omega)$	

- More in the following lectures

Convolution on graphs

- A mathematical operator that computes the “amount of overlap" between two functions

- Convolution in the time domain is equivalent to multiplication in the frequency domain

	Classical convolution	Convolution on graphs	
Time domain	$(f * g)(t) = \int_{-\infty}^{\infty} f(t-\tau) g(\tau) d\tau$		
Frequency domain	$\widehat{(f * g)}(\omega) = \hat{f}(\omega) \cdot \hat{g}(\omega)$	$\widehat{(f * g)}(\lambda) = \big((\chi^T f) \circ \hat{g}\big)(\lambda)$	Frequency/spectral domain

- More in the following lectures

Convolution on graphs

- A mathematical operator that computes the “amount of overlap" between two functions

- Convolution in the time domain is equivalent to multiplication in the frequency domain

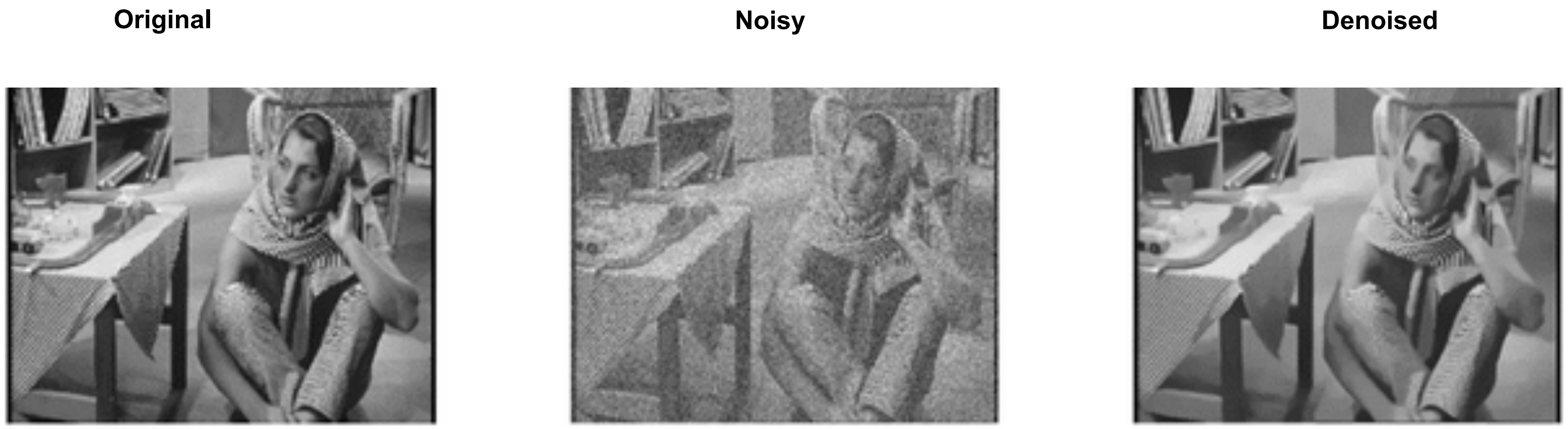
	Classical convolution	Convolution on graphs	
Time domain	$(f * g)(t) = \int_{-\infty}^{\infty} f(t-\tau)g(\tau)d\tau$	$f * g = \chi \hat{g}(\Lambda) \chi^T f = \hat{g}(L) f$	Vertex domain
	$\downarrow$	$\uparrow$	
Frequency domain	$\widehat{(f * g)}(\omega) = \hat{f}(\omega) \cdot \hat{g}(\omega)$	$\widehat{(f * g)}(\lambda) = \big((\chi^T f) \circ \hat{g}\big)(\lambda)$	Frequency/spectral domain

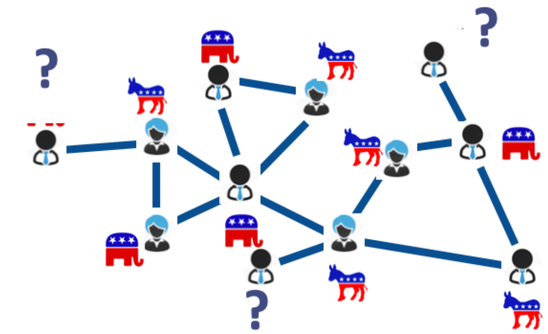
- More in the following lectures

Outline

- Graphs and signals on graphs
- Graph Fourier transform
- Filtering on graphs
- Spectral graph convolution
- **Applications**
  - **Regularization on graphs**
  - **Compression**
  - **Knowledge discovery**

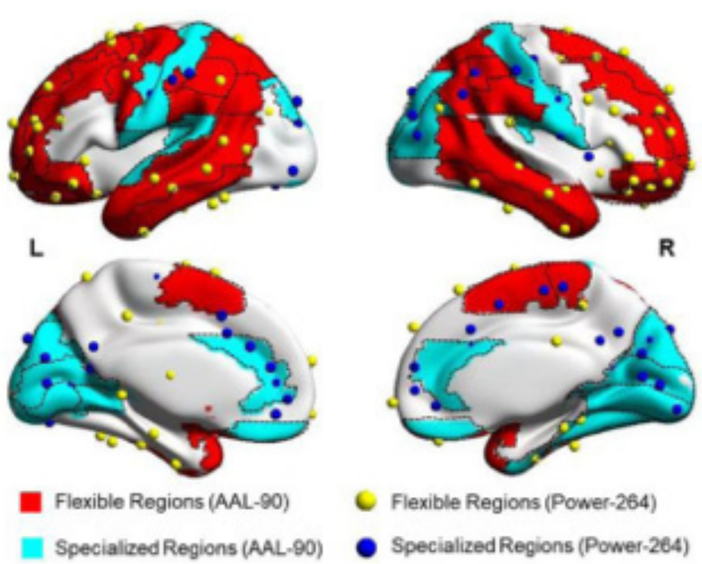
Some typical processing tasks



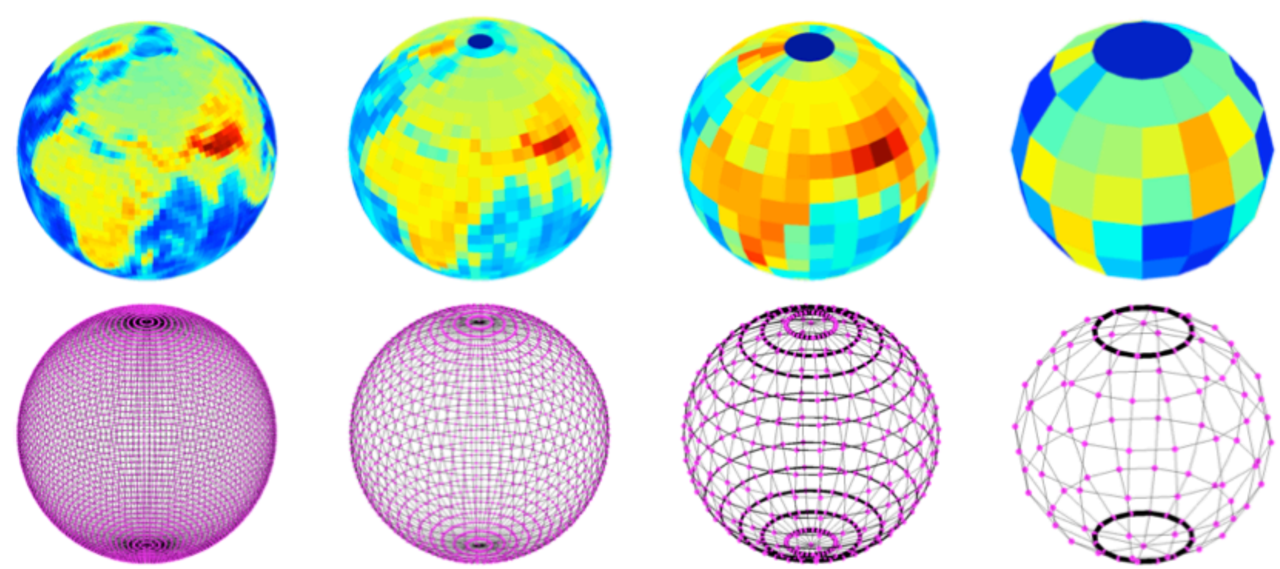


Denoising

Semi-supervised learning

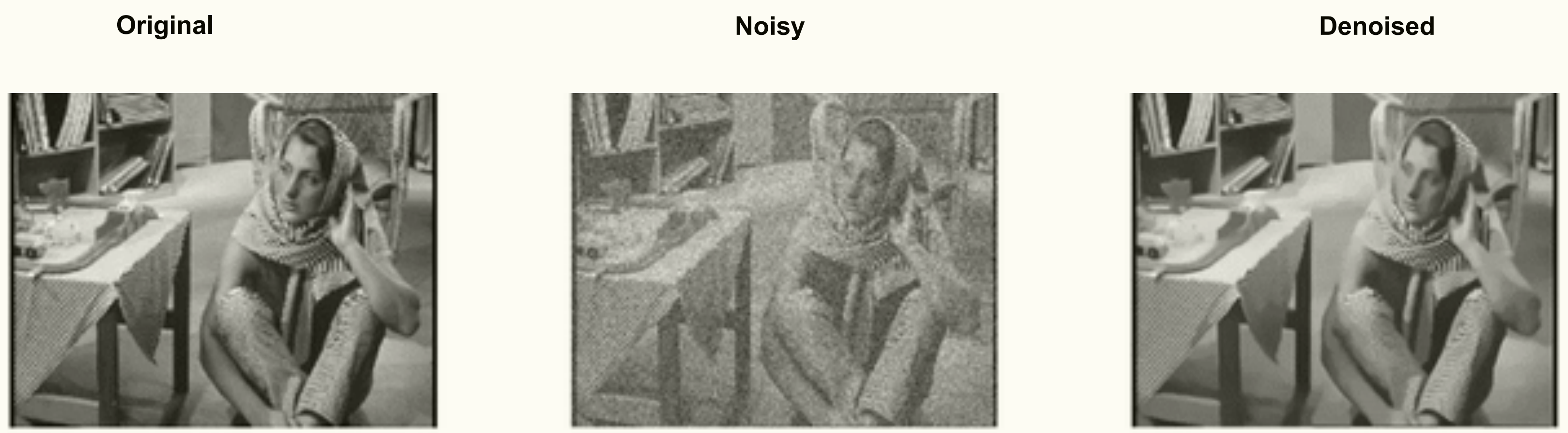


Analysis/Knowledge discovery

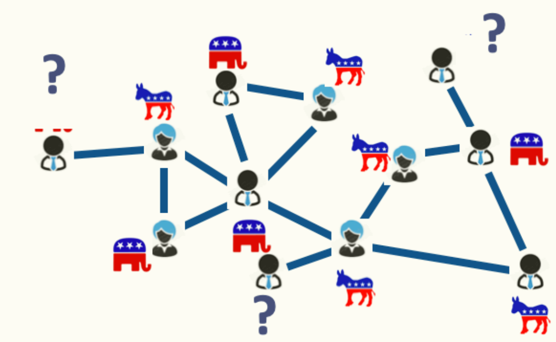


Compression/Visualization

Inverse problems on graphs



Example: Denoising



Example: Semi-supervised learning

- The latent graph signal f generates observed graph signal output y : $f \to y$
- The goal of the inverse problem is to find a mapping such that: $y \to f$
- An inverse problem is inherently underdetermined; Usually **regularized by imposing some prior knowledge** about that data

Regularization on graphs

- Example: **Linear inverse problems on graphs**

$$\tilde{f} = \underset{f}{\operatorname{argmin}} \underbrace{\|y - Mf\|_2^2}_{\text{Fitting term}} + \underbrace{\gamma R(f, G)}_{\text{Regularization term}}$$

- **Fitting term:** Can we recover a graph signal f given some observations y and operator M ?

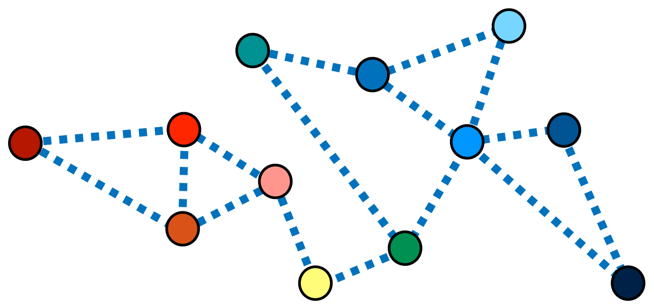
- **Regularization term:** What properties do we expect f to have on the graph?

The graph smoothness prior

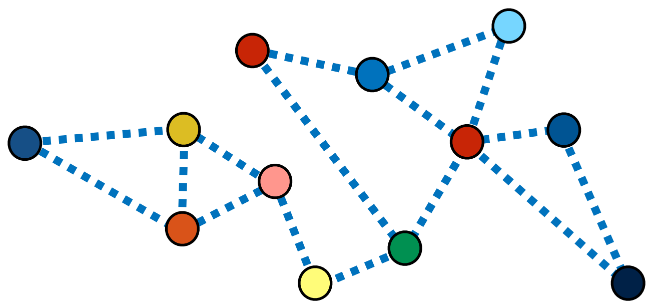
- In many applications, we expect signals to be smooth on the graph
- We recall that:

$$f^T L f = \sum_{n \in \mathcal{V}} \sum_{m \in \mathcal{N}_n} W_{n,m} [f(n) - f(m)]^2$$

- The smaller this quantity, the smoother the signal on that graph
- It is zero iff the signal is constant on the graph



$f_1^T L f_1 = 0.15$



$f_2^T L f_2 = 1.8$

Application: Graph signal denoising

- We observe a noisy graph signal $y = f + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma)$
- The observation matrix is

$$M = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & & \ddots & \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

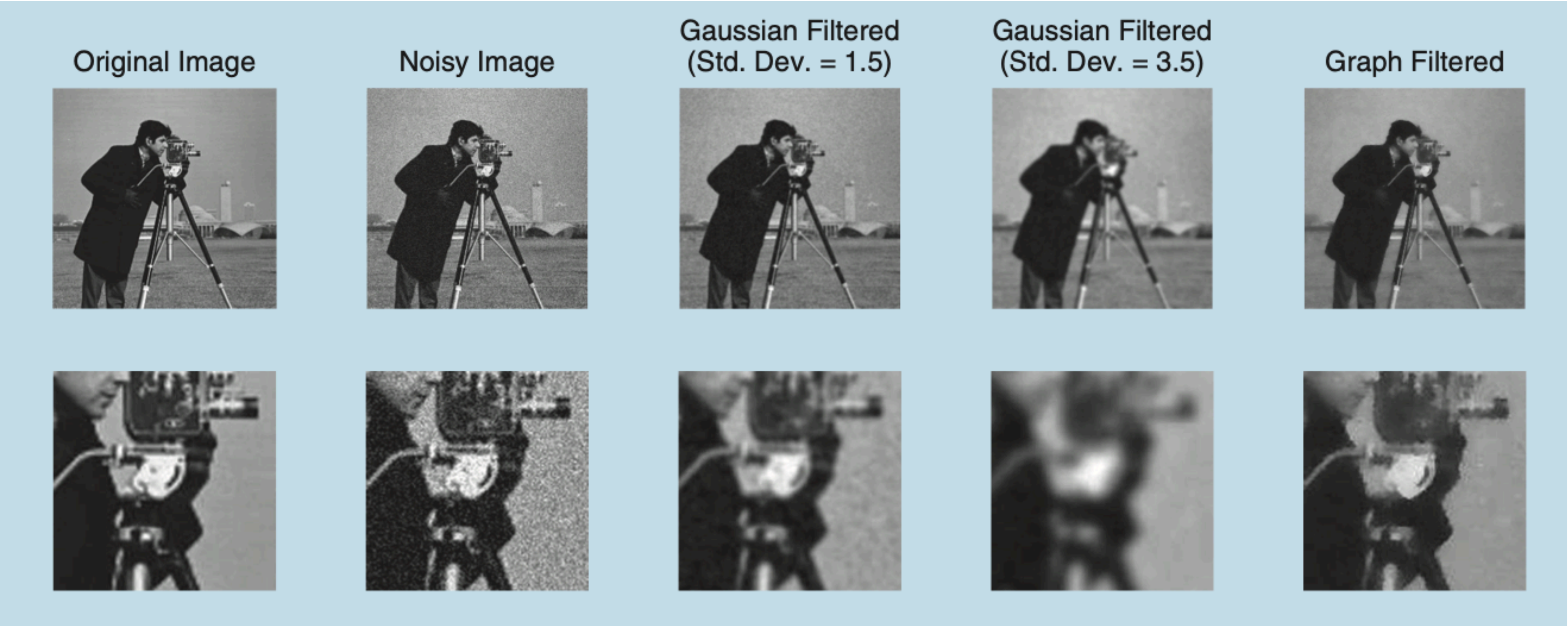
- We wish to recover f by enforcing that it is smooth with respect to the graph

$$\tilde{f} = \underset{f}{\operatorname{argmin}} \|f - y\|_2^2 + \gamma f^T L f$$

- Also known as graph Tikhonov regularization

Application: Image denoising

- Construct a graph that encodes pixel similarity
- Denoise the image by assuming smoothness on the graph



A filtering viewpoint

- Graph regularization can be interpreted as filtering on the graph

$$\tilde{f} = \underset{f}{\operatorname{argmin}} \|f - y\|_2^2 + \gamma f^T L f$$

$$\Downarrow$$

First order gradient

$$\tilde{f} = (I + \gamma L)^{-1} y$$

$$\Downarrow$$

Eigendecomposition of the Laplacian

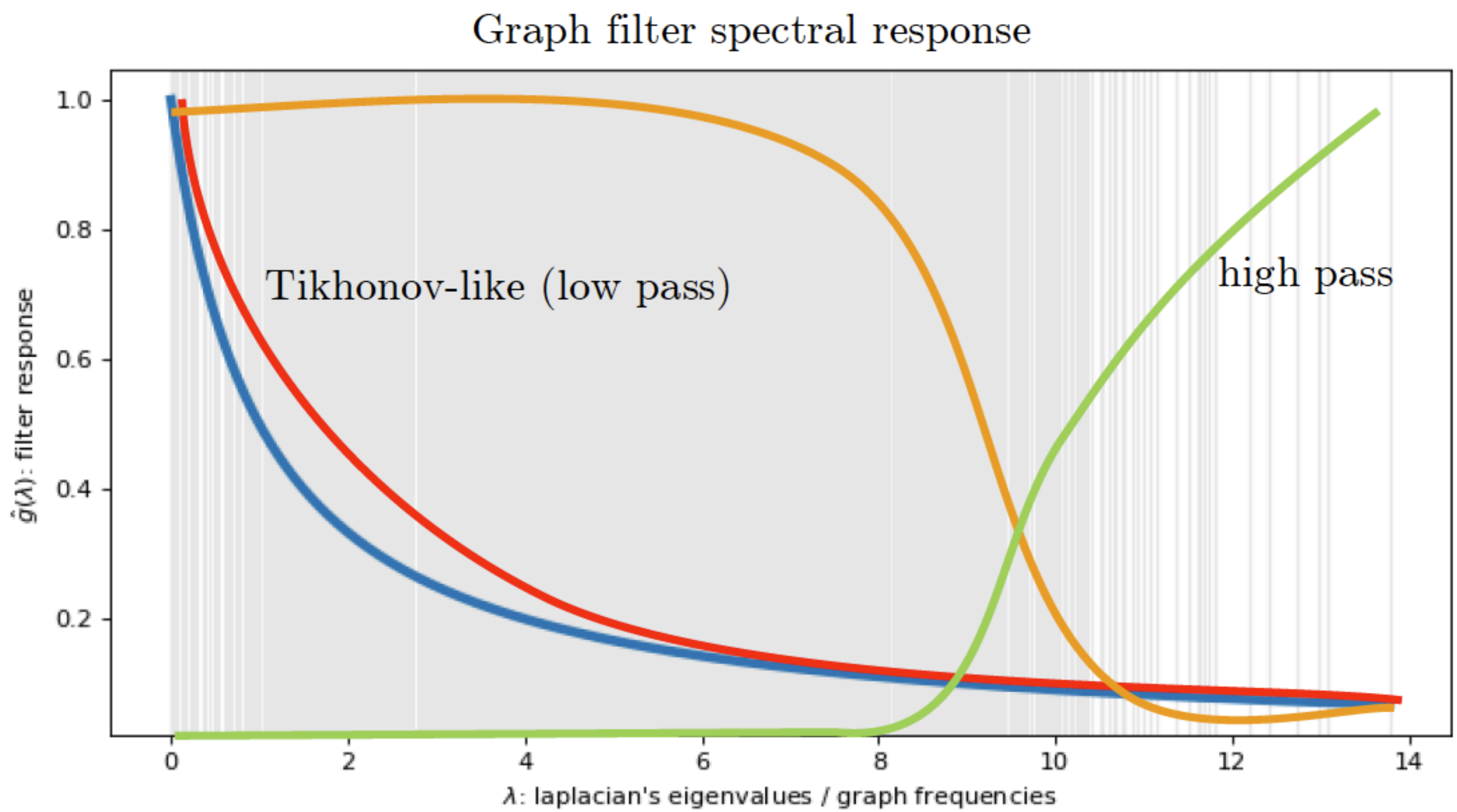
$$\tilde{f} = \chi \boxed{(I + \gamma \Lambda)^{-1}} \chi^T y$$

$$\Downarrow$$

$$g(L)$$

Remove noise by lowpass filtering in the graph spectral domain!

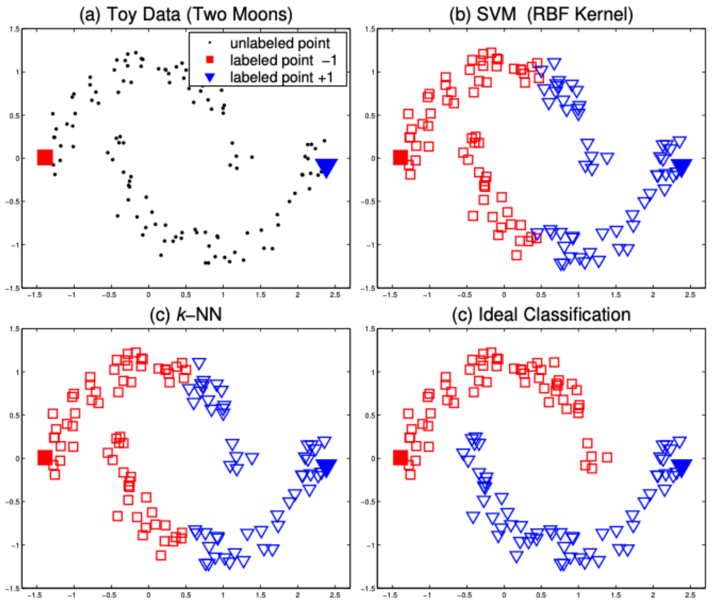
Other graph filters



Application: Semi-supervised learning

- Find missing labels by using information from both labelled and unlabelled data
- Treat labels as a signal on the graph
- Similar nodes on high density regions of the graph should have similar labels

$$\tilde{f} = \underset{f}{\operatorname{argmin}} \|f - y\|_2^2 + \gamma \sum_{n,m} W_{n,m} \| \frac{1}{D_{nn}} f_n - \frac{1}{D_{mm}} f_m \|_2^2$$



Zhou et al., "Learning with Local and Global Consistency", NIPS, 2003

Other regularizers

$$\tilde{f} = \underset{f}{\operatorname{argmin}} \|y - Mf\|_2^2 + \gamma R(f, G)$$

- Discrete p-Dirichlet form:

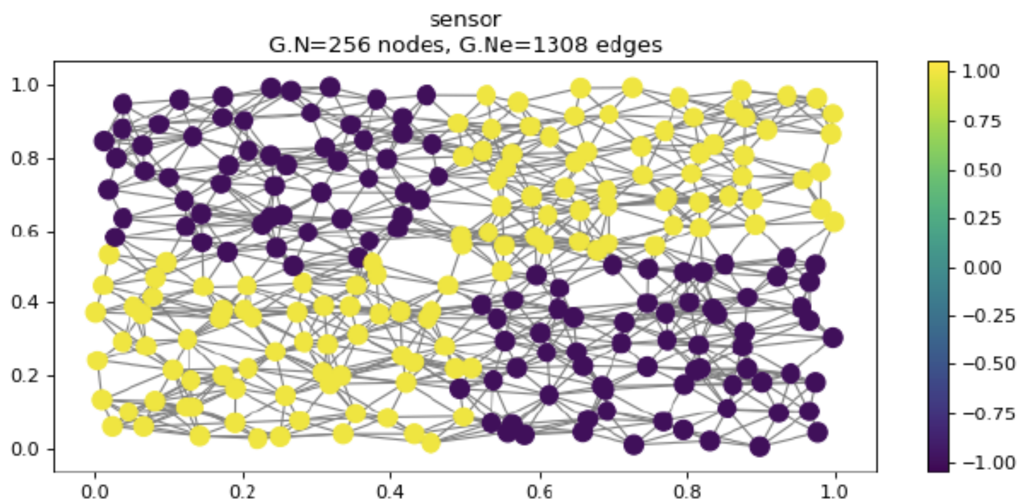
$$R_p(f, G) = \frac{1}{p} \sum_{n \in \mathcal{V}} \|\nabla_n f\|_2^p = \frac{1}{p} \sum_{n \in \mathcal{V}} \Big[\sum_{m \in \mathcal{N}_n} W_{n,m} [f(n) - f(m)]^2 \Big]^{\frac{p}{2}}$$

- Total variation (TV):
  - Promote piecewise smooth signals: $p = 1$

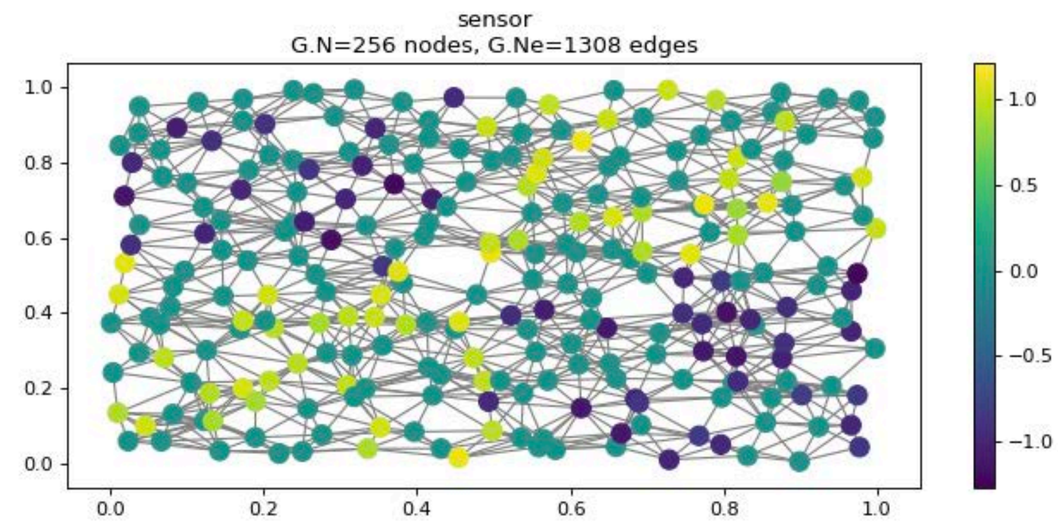
- Sparsity in the graph Fourier basis:
  - Promote a graph signal with only a few non-zero GFT coefficients

$$R(f, G) = \|f\|_1, \quad M = \chi$$

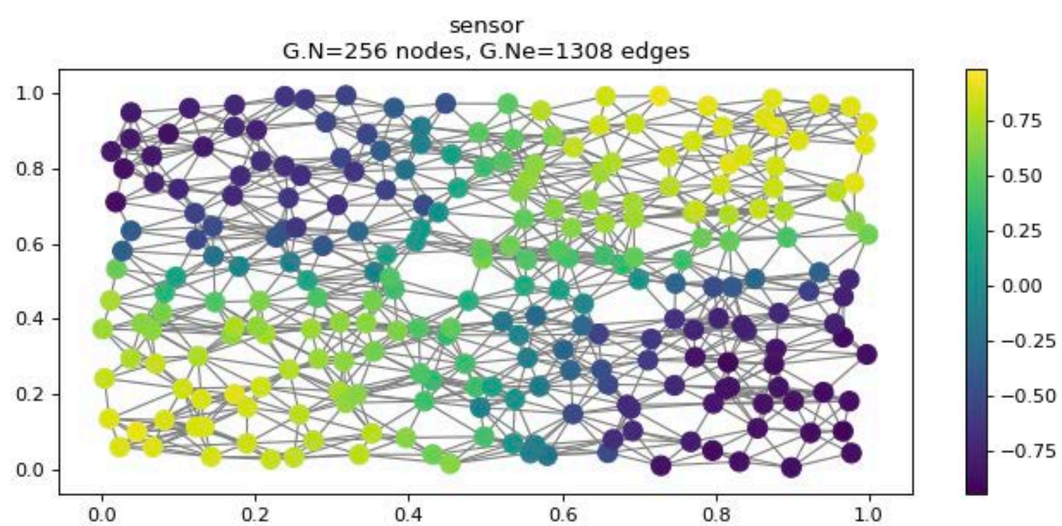
Difference between Tikhonov and TV



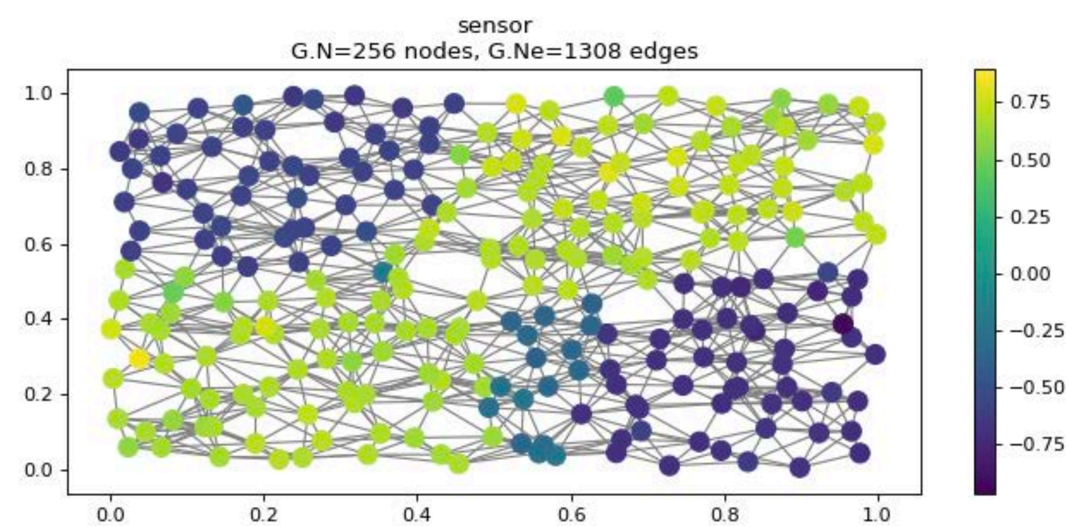
Ground-truth signal



Noisy signal



Denoised signal - Tikhonov

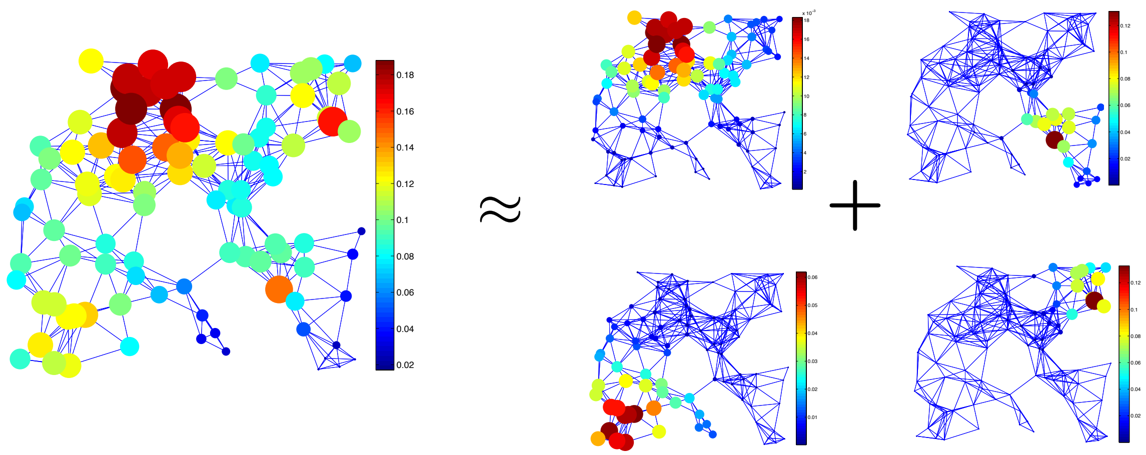


Denoised signal - TV

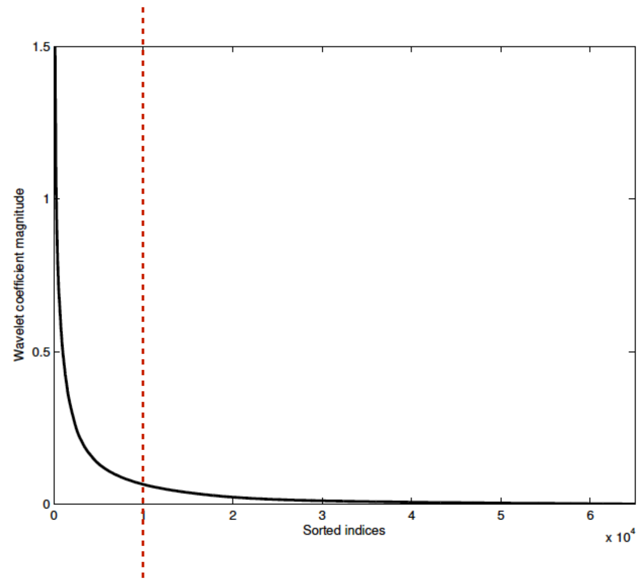
https://pygsp.readthedocs.io/en/stable/tutorials/optimization.html

Application: Compression

- Desirable: Capture a large part of the signal with a few coefficients



- Typically performed by projecting the data in a domain/transform where the signal is compressible or sparse

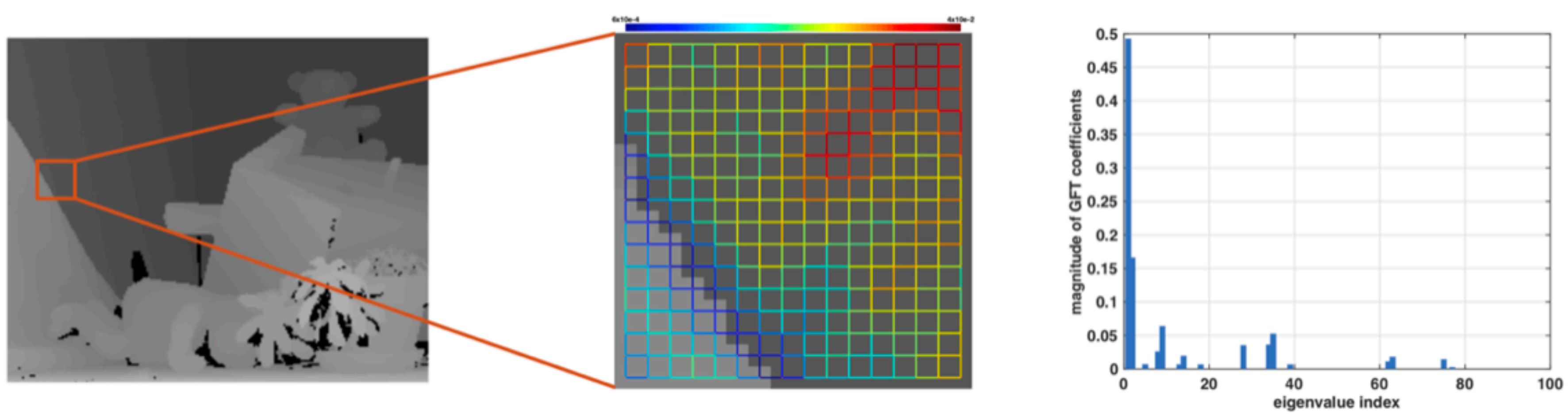


Original image **Most significant coefficients** **Reconstructed image**

Application: Image compression

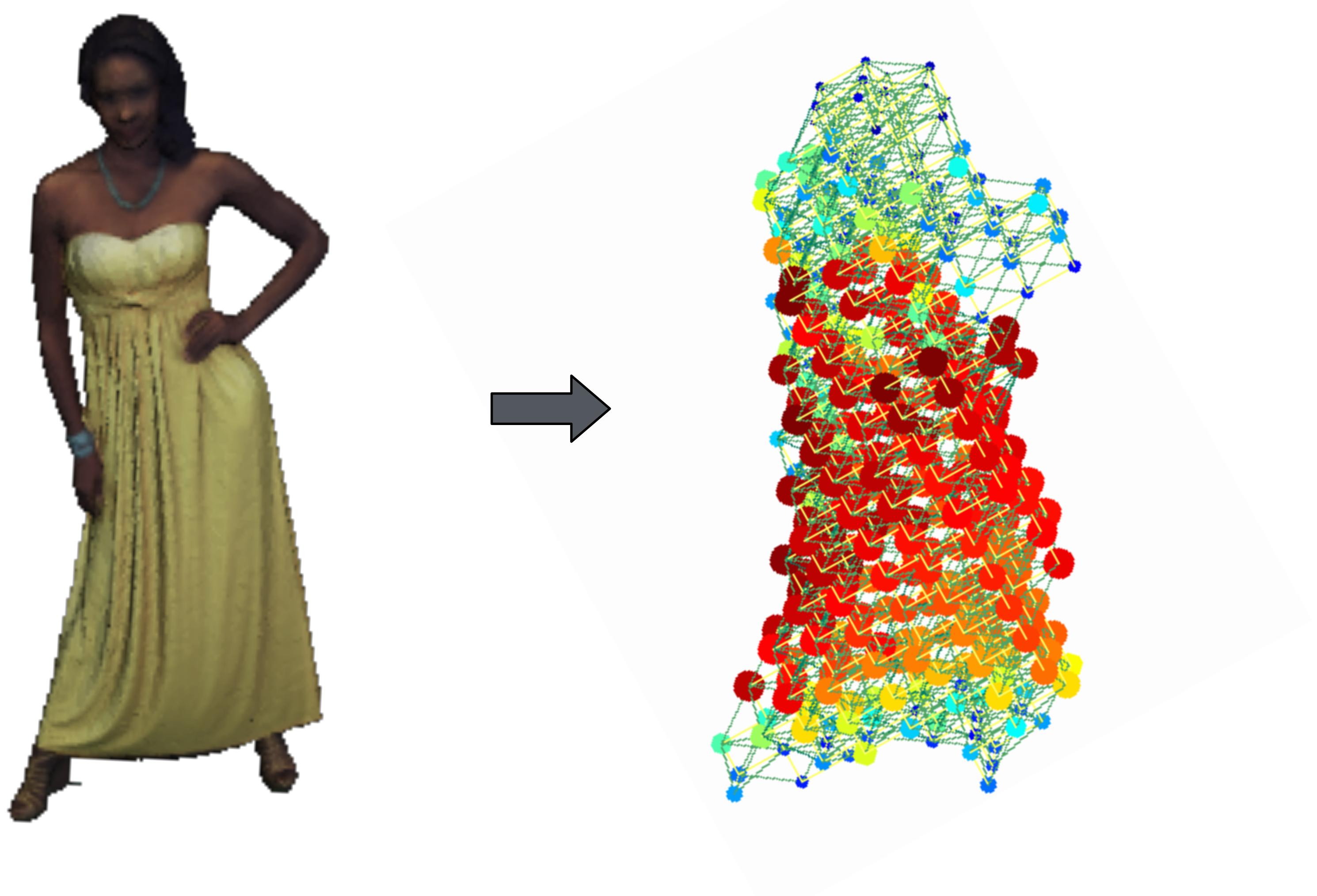
- The graph Fourier transform has been used to compress smooth signals on the graph
  - Intuition: Main energy is concentrated in the first GFT coefficients

- Example: image compression
  - Construct a graph that encodes pixel similarity
  - The graph Fourier transform has been used as an alternative to classical transforms



Application: 3D point clouds compression

- Graphs provide a way to represent point clouds



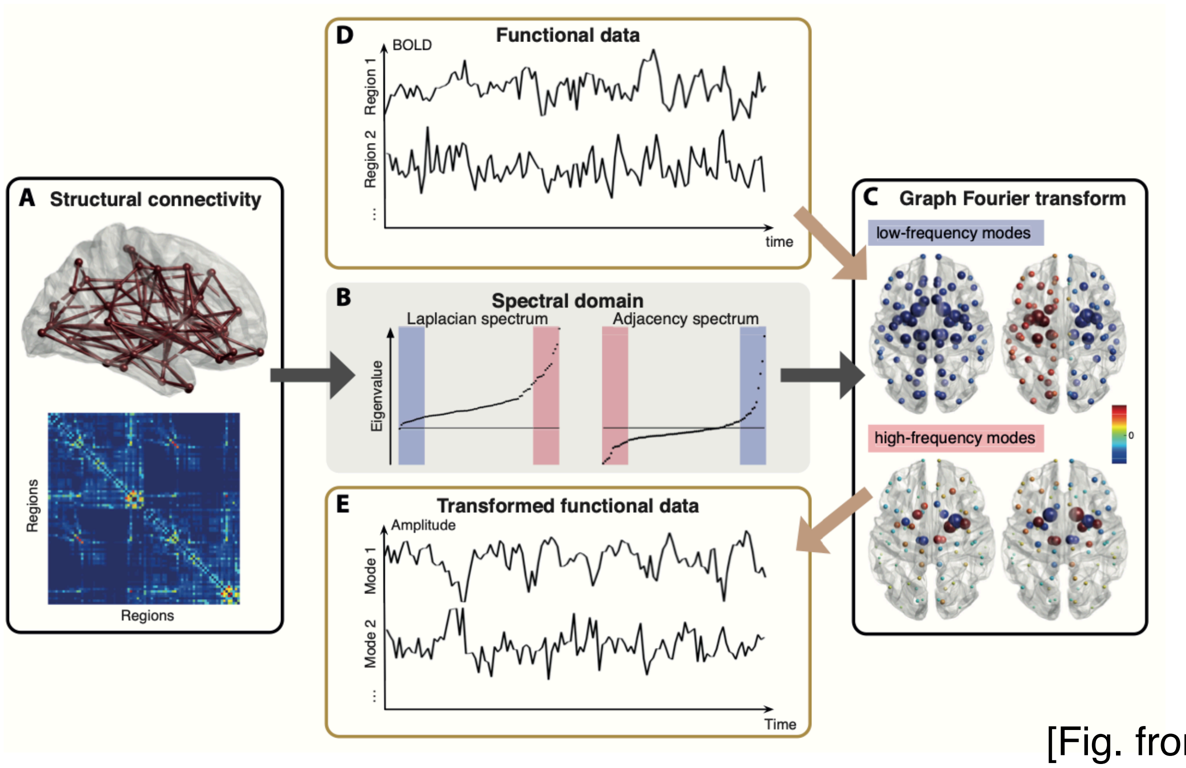
- The graph Fourier transform has been used to capture large parts of the cooler attributes with a few coefficients

Zhang et al., “Point cloud attribute compression with graph transforms”, ICIP, 2014

Knowledge discovery: Neuroscience

- Graph based transforms have been successful in domain specific knowledge discovery

- In neuroscience, GSP tools have been used to improve our understanding of the biological mechanisms underlying human cognition and brain disorders

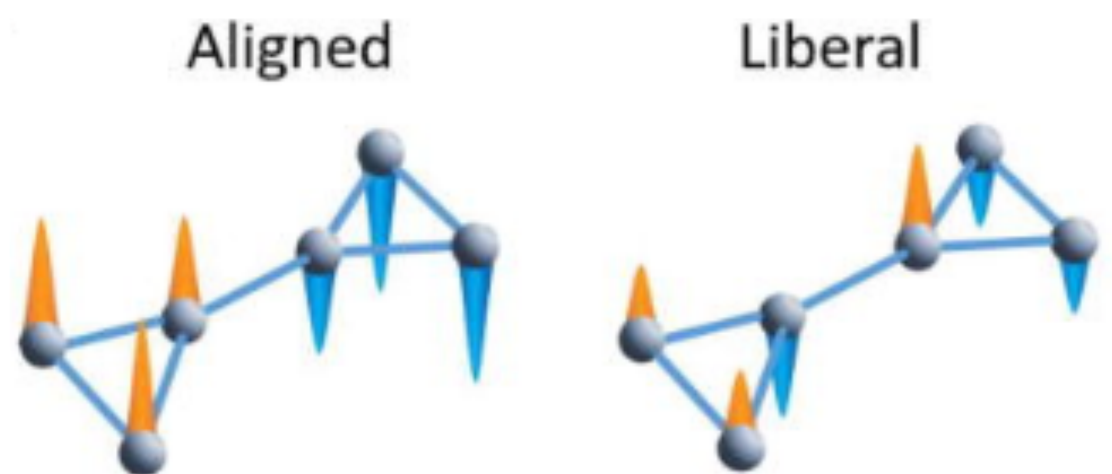


[Fig. from Huang'18]

- Analysis in the spectral domain reveals the variation of signals on the anatomical network

Graph spectral analysis for understanding cognitive flexibility

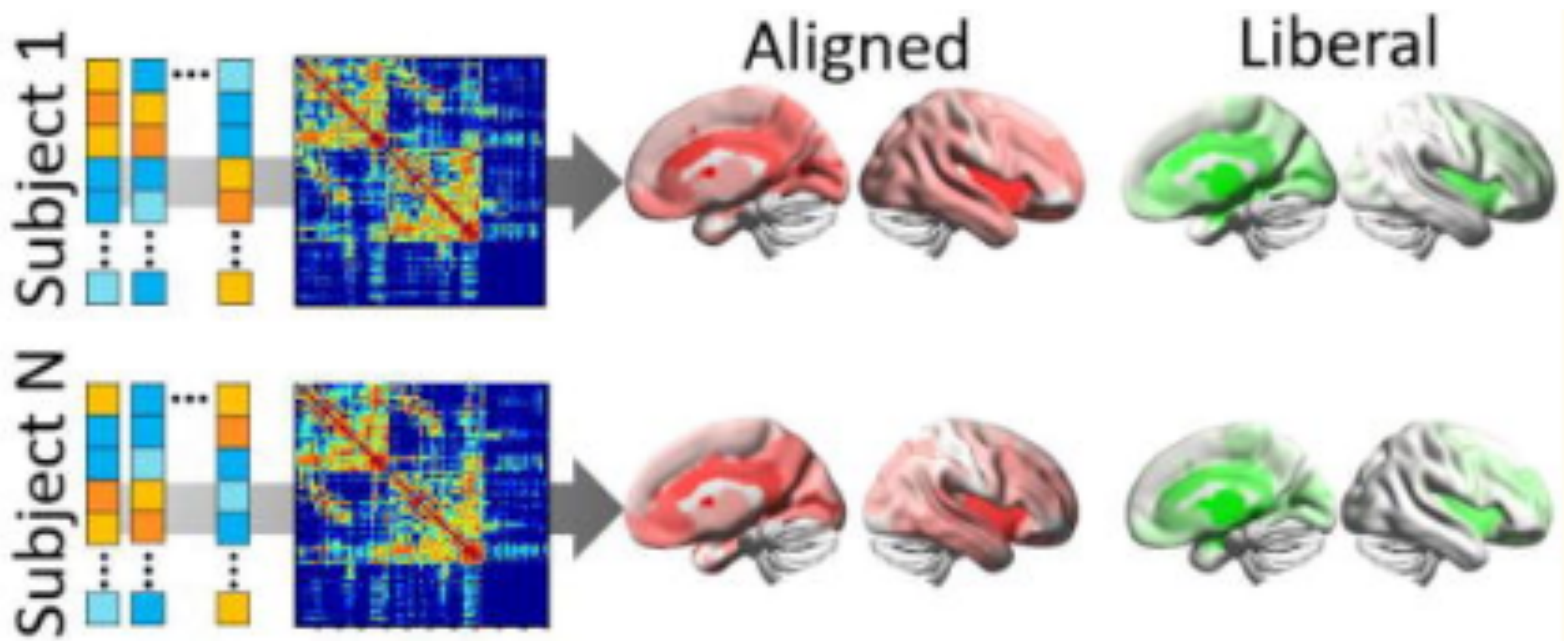
- Cognitive flexibility describes the human ability to switch between modes of mental function
- Integrating brain network structure, function, and cognitive measures is key
- GFT allows to decompose each BOLD signal into two components
  - Aligned: Component of the signal that is aligned with the anatomical network
  - Liberal: Component of the signal that does not align with the anatomical network



Medaglia et al., "Functional Alignment with Anatomical Networks is Associated with Cognitive Flexibility", Nat. Hum. Behav., 2018

BOLD signal alignment across the brain

- Functional alignment with anatomical networks facilitates cognitive flexibility (lower switch costs)
  - Liberal signals are concentrated in subcortical regions and cingulate cortices
  - Aligned signals are concentrated in subcortical, default mode, fronto-parietal, and cingulo-opercular systems



BOLD signals **White matter network** **Decomposition of the signals into aligned and liberal using GFT**

Summary

- Graphs are natural tools to capture the data domain

- Going beyond graph structure implies understanding the interplay between that domain and the data:
  - Jointly consider domain (i.e., graph) and data (i.e., graph signals) that live in that domain

- Some key concepts can be directly generalized from regular grids to graphs
  - Tranforms on graph
  - Filtering on graph
  - Convolution on graph (more in the following lectures…)

- Many applications including network analysis, denoising, compression

References

- The Emerging Field of Signal Processing on Graphs, Shuman et al., 2013
- Graph Signal Processing, Ortega et al., 2018
- Toolbox: https://pygsp.readthedocs.io/en/stable/index.html