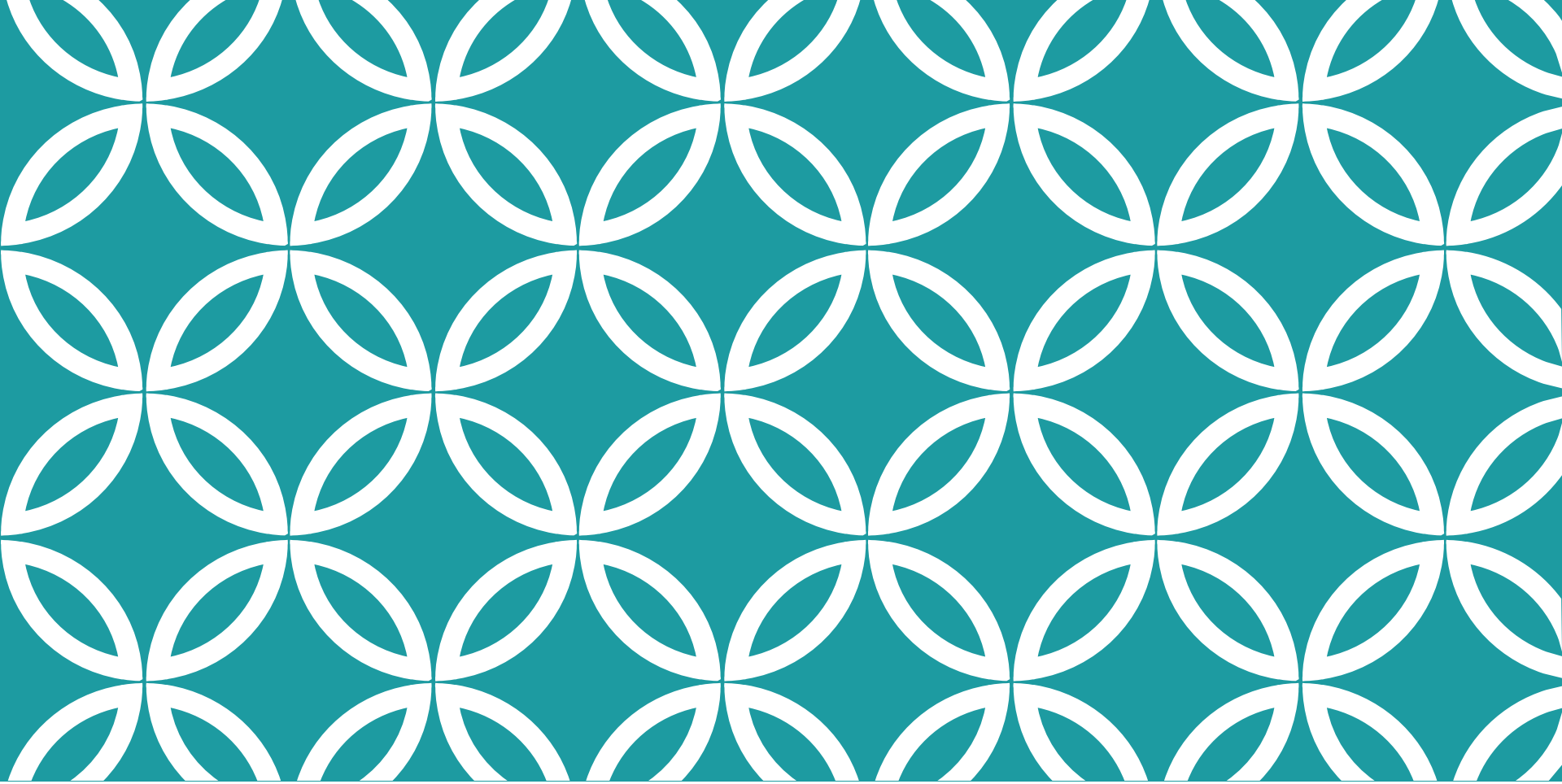


GRAPH SIGNAL PROCESSING

GSP

Research in Graph Signal Processing (GSP) aims to develop tools for processing data defined on irregular graph domains.

PURPOSE

The internet of things by definition is multi-device/thing networks, where the devices (things) of a network collect, process and exchange data in a continuous fashion.

The ability to exchange data between the network devices facilitates complementary and competitive collaboration among the devices of IOT networks.

In many cases, the data captured by the network in totality provide an insightful view at the IOT network level.

Such view should answer questions such as: which ”thing” is more tightly connected to what other thing?, how active is the network? How much connected is the network? How the data collected by the network vary in time, at the individual “thing" level as well as at a multi-thing level, and the network level.

SIGNAL PROCESSING TOOLS

The domain of signal processing provide a suite of signal processing tools, e.g., FFT, etc, that are quite effective in analyzing time (space) varying signals.

These tools have been well utilized in analyzing single variable and regular multi-variable time (space) signals.

The question is how one can adopt these tools in irregular multi-variable situations, such as IOT networks.

GRAPH REPRESENTATION OF IOT

Graphs are a powerful tool for capturing an IOT network structure and the interconnectivity among its “things”, to the extent it can be used to facilitate a wide range of analysis of the network and its “things”.

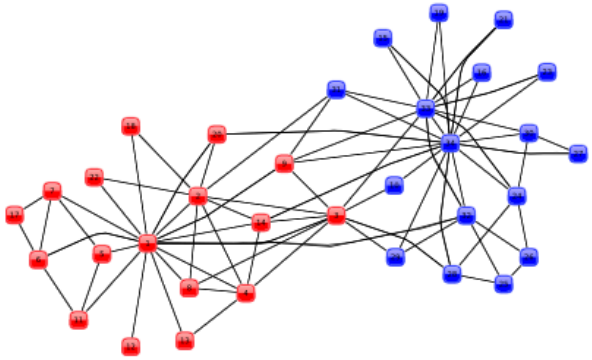
It is a powerful abstraction that offers a wide range of Graph representation and manipulation tools, including the recent development of Graph Signal Processing.

GRAPH SIGNAL PROCESSING

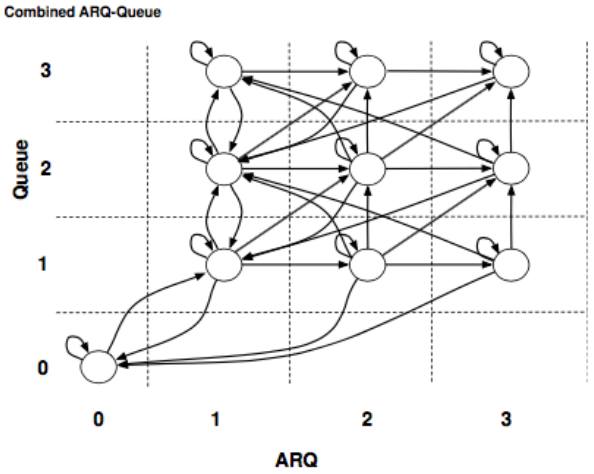
Graph Signal Processing (GSP) aims at developing tools for processing data defined on irregular graph domains.

It is concerned with a wide variety of operations on graphs, ranging from simple ones like frequency analysis and filtering to advanced ones like interpolation or graph learning.

- Examples in non-Euclidean settings



(a)

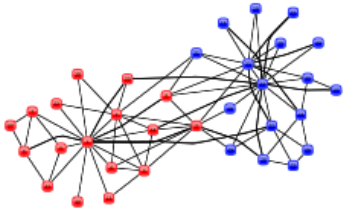


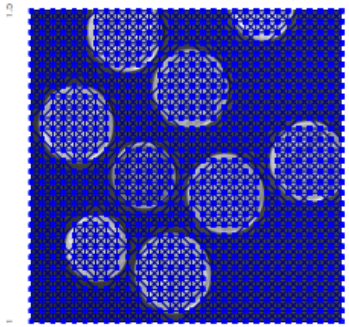
(b)

(a) Social Networks [4], (b) Finite State Machines(FSM)

Graphs can capture complex relational characteristics (e.g., spatial, topological).

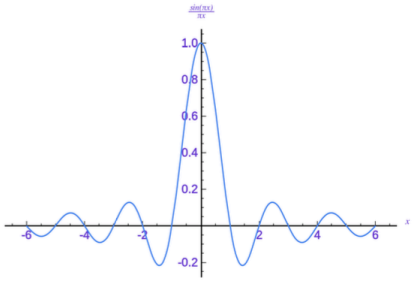
EXAMPLES

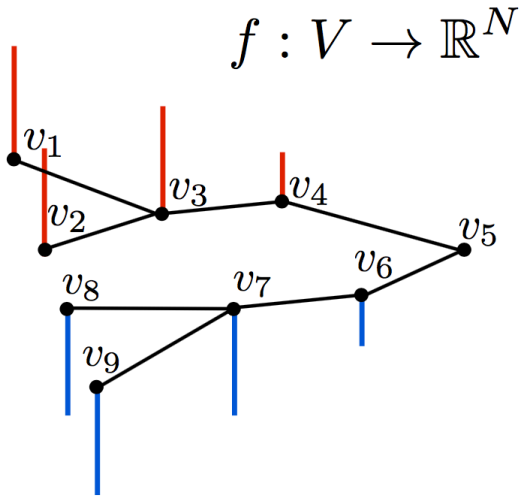




- Sensor network
  - Relative positions of sensors (kNN), temperature
  - does temperature vary smoothly?
- Social network
  - friendship relationship, age
  - are friends of similar age?
- Images
  - pixel positions and similarity, pixel values
  - discontinuities and smoothness

REGULAR VS IRREGULAR

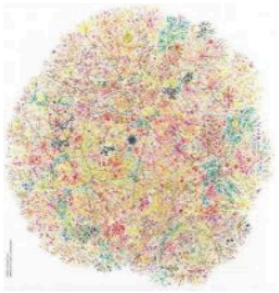




How to generalize classical signal processing tools on irregular domains such as graphs?

Online social media

Internet

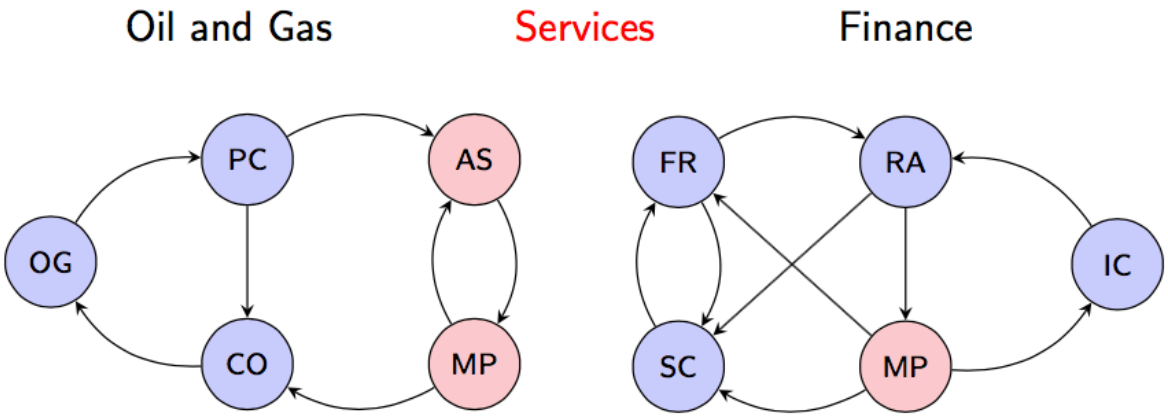


Clean energy and grid analytics

- Desiderata: Process, analyze and learn from network data [Kolaczyk'09]
- Network as graph $G = (\mathcal{V}, \mathcal{E})$: encode pairwise relationships
- Interest here not in G itself, but in data associated with nodes in $\mathcal{V}$
  $\Rightarrow$ Object of study is a graph signal $\mathbf{x}$
- Q: Graph signals common and interesting as networks are?

NETWORK OF ECONOMIC SECTORS USA

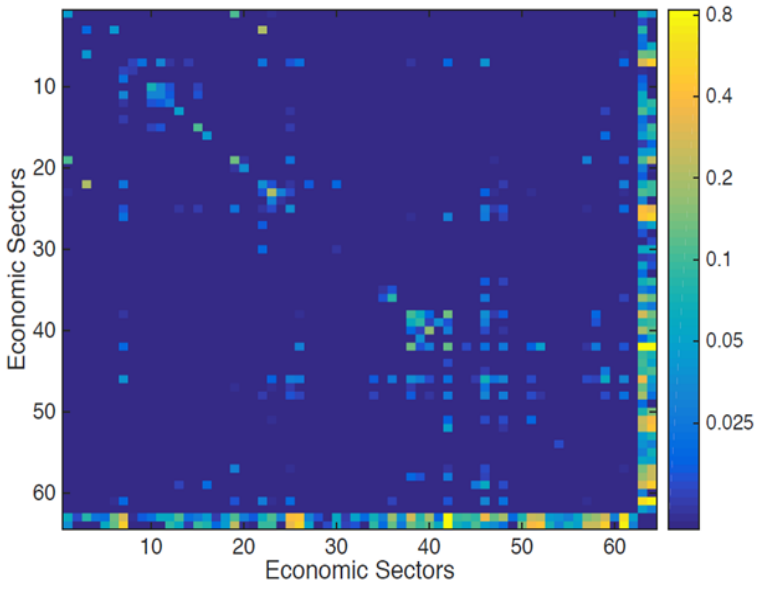
- Bureau of Economic Analysis of the U.S. Department of Commerce
- $\mathcal{E}$ = Output of sector i is an input to sector j (62 sectors in $\mathcal{V}$)



- Oil extraction (OG), Petroleum and coal products (PC), Construction (CO)
- Administrative services (AS), Professional services (MP)
- Credit intermediation (FR), Securities (SC), Real state (RA), Insurance (IC)
- Only interactions stronger than a threshold are shown

US NETWORK OF ECONOMICS SECTORS

- Bureau of Economic Analysis of the U.S. Department of Commerce
- $\mathcal{E}$ = Output of sector i is an input to sector j (62 sectors in $\mathcal{V}$)

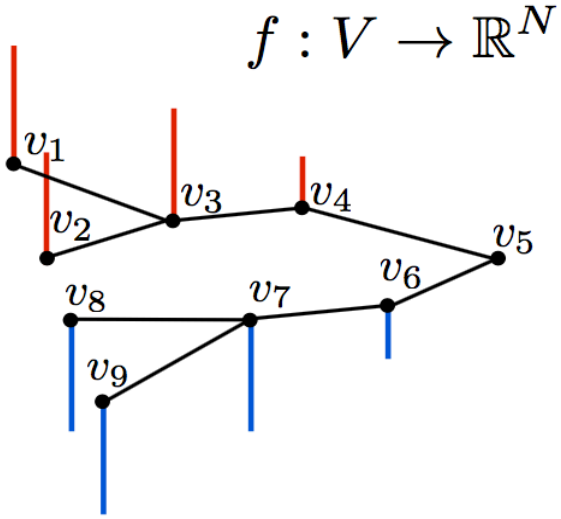


- A few sectors have widespread strong influence (services, finance, energy)
- Some sectors have strong indirect influences (oil)
- The heavy last row is final consumption

- This is an interesting network $\Rightarrow$ Signals on this graph are as well

Graph signal processing

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



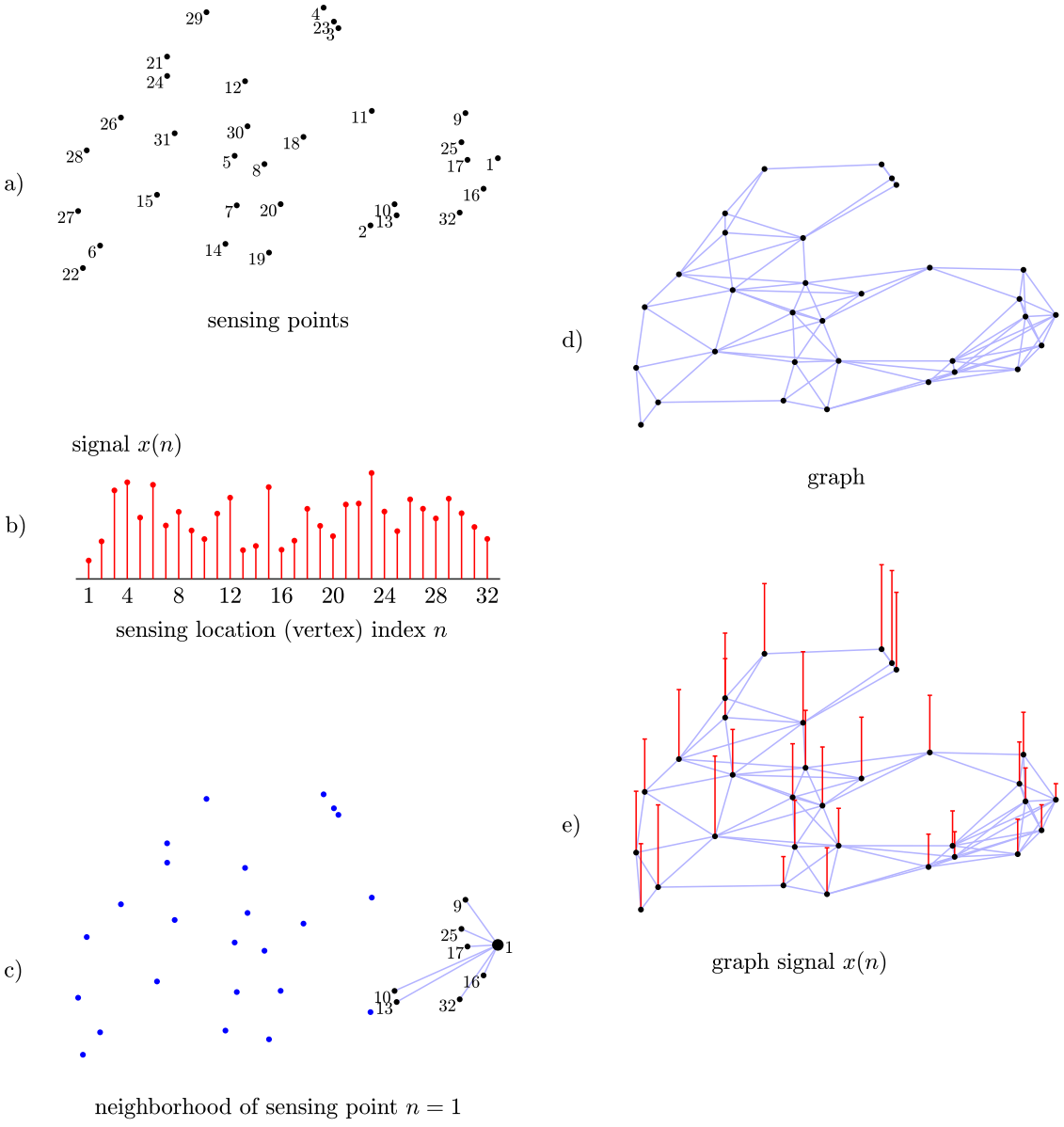
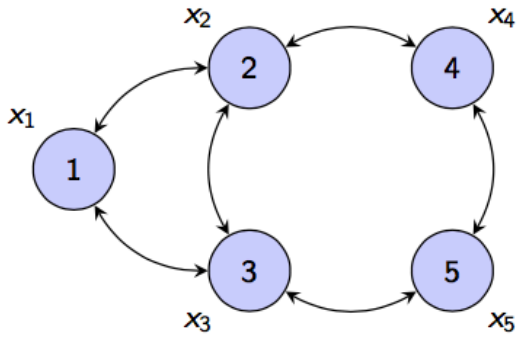


Fig. 1 Graph and a signal on the graph illustration.

GRAPH SIGNAL PROCESSING

- Graph SP: broaden classical SP to graph signals [Shuman et al.'13]
  $\Rightarrow$ Our view: GSP well suited to study network (diffusion) processes

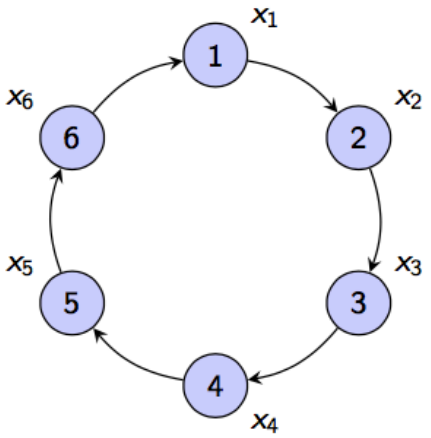


- **As.:** Signal properties related to topology of G (locality, smoothness)
  $\Rightarrow$ Algorithms that fruitfully leverage this relational structure
- Q: Why do we expect the graph structure to be useful in processing $\mathbf{x}$?

SIGNAL STRUCTURE

- Signal and Information Processing is about exploiting signal structure

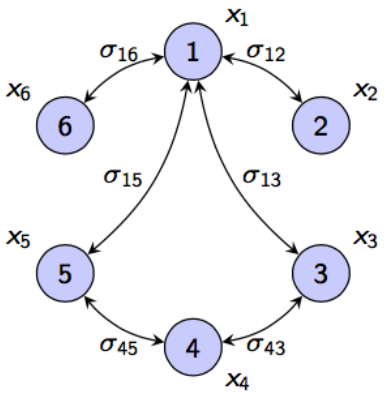
- Discrete time described by cyclic graph
  - $\Rightarrow$ Time n follows time $n-1$
  - $\Rightarrow$ Signal value x_n similar to x_{n-1}
- Formalized with the notion of frequency



- Cyclic structure $\Rightarrow$ Fourier transform $\Rightarrow \tilde{\mathbf{x}} = \mathbf{F}^H \mathbf{x}$ $\left(F_{kn} = \frac{e^{j2\pi kn/N}}{\sqrt{N}} \right)$
- Fourier transform $\Rightarrow$ Projection on eigenvector space of cycle

COVARIANCE AND PRINCIPLE COMPONENTS

- Random signal with mean $\mathbb{E}[\mathbf{x}] = 0$ and covariance $\mathbf{C}_x = \mathbb{E}\left[\mathbf{x}\mathbf{x}^H\right]$
  $\Rightarrow$ Eigenvector decomposition $\mathbf{C}_x = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^H$
- Covariance matrix $\mathbf{C}_x$ is a graph
  $\Rightarrow$ Not a very good graph, but still
- Precision matrix $\mathbf{C}_x^{-1}$ a common graph too
  $\Rightarrow$ Conditional dependencies of Gaussian $\mathbf{x}$

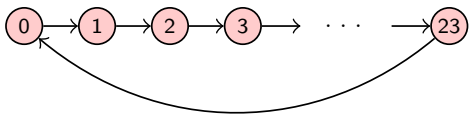


- Covariance matrix structure $\Rightarrow$ Principal components (PCA) $\Rightarrow \tilde{\mathbf{x}} = \mathbf{V}^H\mathbf{x}$
- PCA transform $\Rightarrow$ Projection on eigenvector space of (inverse) covariance
- Q: Can we extend these principles to general graphs and signals?

GRAPHS

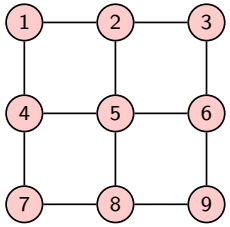
- Formally, a graph G (or a network) is a triplet $(\mathcal{V}, \mathcal{E}, W)$
- $\mathcal{V} = \{1, 2, \ldots, N\}$ is a finite set of N nodes or vertices
- $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of edges defined as ordered pairs (n, m)
  - Write $\mathcal{N}(n) = \{m \in \mathcal{V} : (m, n) \in \mathcal{E}\}$ as the in-neighbors of n
- $W : \mathcal{E} \to \mathbb{R}$ is a map from the set of edges to scalar values w_{nm}
  - Represents the level of relationship from n to m
  - Often weights are strictly positive, $W : \mathcal{E} \to \mathbb{R}_{++}$
- Unweighted graphs $\Rightarrow w_{nm} \in \{0, 1\}$, for all $(n, m) \in \mathcal{E}$
- Undirected graphs $\Rightarrow (n, m) \in \mathcal{E}$ if and only if $(m, n) \in \mathcal{E}$ and $w_{nm} = w_{mn}$, for all $(n, m) \in \mathcal{E}$

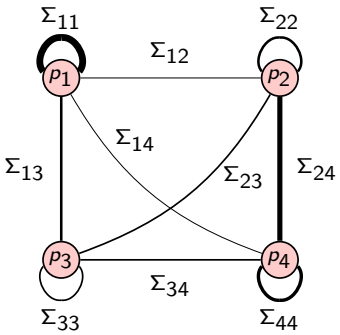
GRAPH EXAMPLES



- Unweighted and directed graphs (e.g., time)
  - $\mathcal{V} = \{0, 1, \ldots, 23\}$
  - $\mathcal{E} = \{(0,1), (1,2), \ldots, (22,23), (23,0)\}$
  - $W : (n, m) \mapsto 1$, for all $(n, m) \in \mathcal{E}$

- Unweighted and undirected graphs (e.g., image)
  - $\mathcal{V} = \{1, 2, 3, \ldots, 9\}$
  - $\mathcal{E} = \{(1,2), (2,3), \ldots, (8,9), (1,4), \ldots, (6,9)\}$
  - $W : (n, m) \mapsto 1$, for all $(n, m) \in \mathcal{E}$





- Weighted and undirected graphs (e.g., covariance)
  - $\mathcal{V} = \{1, 2, 3, 4\}$
  - $\mathcal{E} = \{(1,1), (1,2), \ldots, (4,4)\} = \mathcal{V} \times \mathcal{V}$
  - $W : (n, m) \mapsto \sigma_{nm} = \sigma_{mn}$, for all (n, m)

ADJACENCY MATRIX

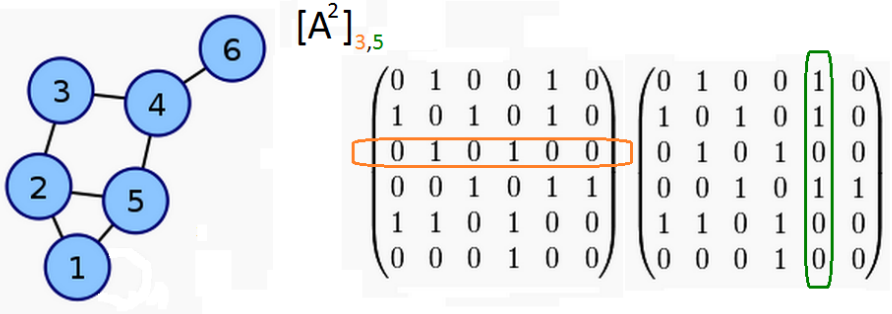
- Algebraic graph theory: matrices associated with a graph G
  - $\Rightarrow$ Adjacency **A** and Laplacian **L** matrices
  - $\Rightarrow$ Spectral graph theory: properties of G using spectrum of **A** or **L**
- Given $G = (\mathcal{V}, \mathcal{E}, W)$, the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ is

$$A_{nm} = \begin{cases} w_{nm}, & \text{if } (n,m) \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases}$$

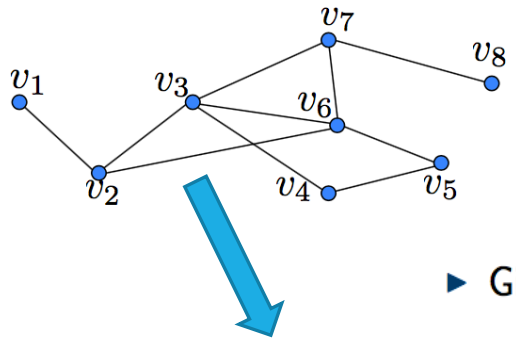
- Matrix representation incorporating all information about G
  - $\Rightarrow$ For unweighted graphs, positive entries represent connected pairs
  - $\Rightarrow$ For weighted graphs, also denote proximities between pairs

DEGREE AND K-HOP NEIGHBOURS

- If G is unweighted and undirected, the degree of node i is $|\mathcal{N}(i)|$
  - $\Rightarrow$ In directed graphs, have out-degree and an in-degree
- Using the adjacency matrix in the undirected case
  - $\Rightarrow$ For node i: $\deg(i) = \sum_{j \in \mathcal{N}(i)} A_{ij} = \sum_j A_{ij}$
  - $\Rightarrow$ For all N nodes: $\mathbf{d} = \mathbf{A1} \rightarrow$ Degree matrix: $\mathbf{D} := \text{diag}(\mathbf{d})$
- Q: Can this be extended to k-hop neighbors? $\rightarrow$ Powers of $\mathbf{A}$
  - $\Rightarrow$ $[\mathbf{A}^k]_{ij}$ non-zero only if there exists a path of length k from i to j
  - $\Rightarrow$ Support of $\mathbf{A}^k$: pairs that can be reached in k hops



Graph Laplacian



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

Weighted and undirected graph:

$G = \{V, E\}$

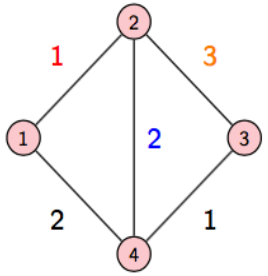
- Given undirected G with $\mathbf{A}$ and $\mathbf{D}$, the Laplacian matrix $\mathbf{L} \in \mathbb{R}^{N \times N}$ is

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

$\Rightarrow$ Equivalently, $\mathbf{L}$ can be defined element-wise as

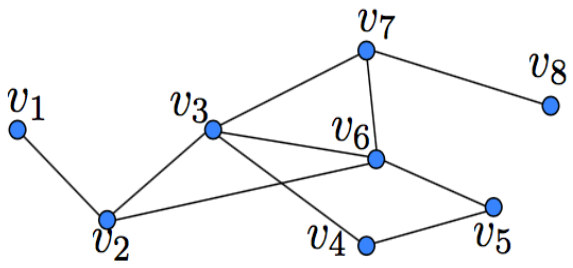
$$L_{ij} = \begin{cases} \deg(i), & \text{if } i = j \\ -w_{ij}, & \text{if } (i,j) \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases}$$

- Normalized Laplacian: $\mathcal{L} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$ (we will focus on $\mathbf{L}$)



$$\mathbf{L} = \begin{bmatrix} 3 & -1 & 0 & -2 \\ -1 & 6 & -3 & -2 \\ 0 & -3 & 4 & -1 \\ -2 & -2 & -1 & 5 \end{bmatrix}$$

Graph Laplacian



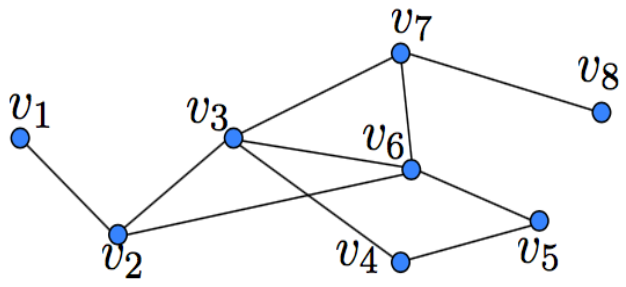
Weighted and undirected graph:

$G = \{V, E\}$

$D = \mathrm{diag}(\mathrm{degree}(v_1) \quad \ldots \quad \mathrm{degree}(v_N))$

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

Graph Laplacian



Weighted and undirected graph:

$G = \{V, E\}$

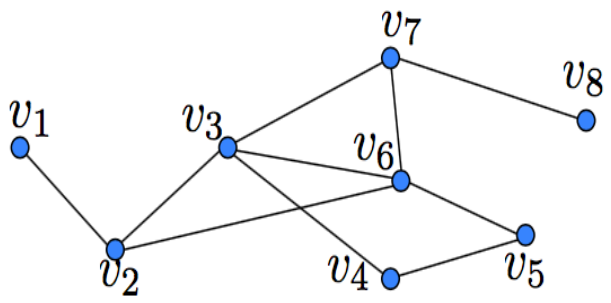
$D = \mathrm{diag}(\mathrm{degree}(v_1) \quad \ldots \quad \mathrm{degree}(v_N))$

$L = D - A$ **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 A L

Graph Laplacian



Weighted and undirected graph:

$G = \{V, E\}$

$D = \mathrm{diag}(\mathrm{degree}(v_1) \quad \ldots \quad \mathrm{degree}(v_N))$

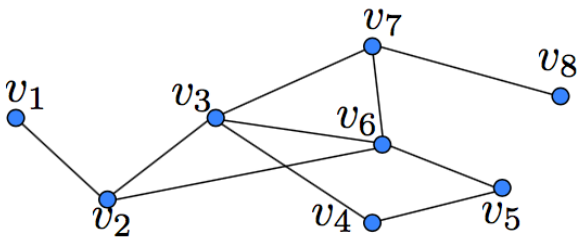
$L = D - A$ **Equivalent to G!**

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

D A L

- Symmetric
- Off-diagonal entries non-positive
- Rows sum up to zero

Graph Laplacian



Weighted and undirected graph:

$G = \{V, E\}$

$D = \mathrm{diag}(\mathrm{degree}(v_1) \quad \ldots \quad \mathrm{degree}(v_N))$

$L = D - A$ **Equivalent to G!**

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

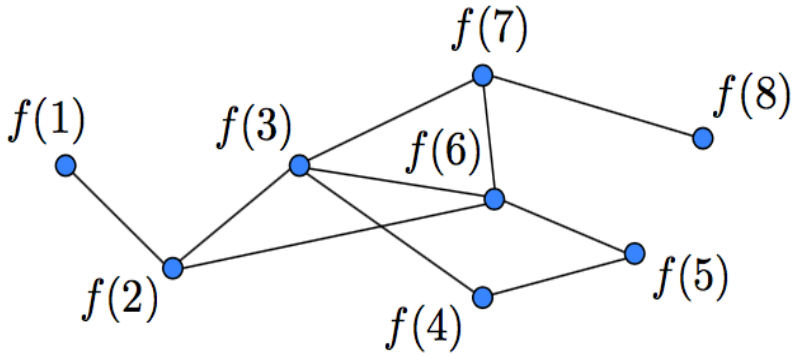
D A L

- Symmetric
- Off-diagonal entries non-positive
- Rows sum up to zero

Why graph Laplacian?

- standard stencil approximation of the Laplace operator
- leads to a Fourier-like transform

Graph Laplacian



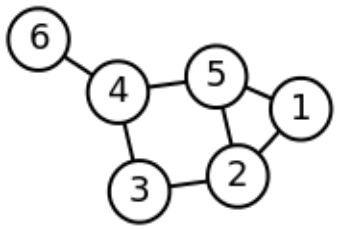
Graph signal $f : V \to \mathbb{R}^N$

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

L

Modeling the graph

Undirected graph



Degree matrix: **D**

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

Adjacency matrix: **A**

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

Laplacian matrix: **L**

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

The **adjacency matrix** is a matrix, $\mathbf{A}$, such that $\mathbf{A}_{ij} = w_{ij}$.

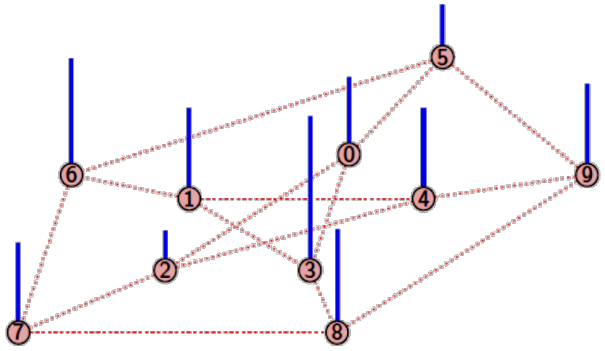
if the graph is undirected, $w_{ij} = w_{ji}$, and $\mathbf{A}$ is symmetric

The **degree matrix** of G is a diagonal matrix, $\mathbf{D}$,
with entries $(\mathbf{D})_{ii} = \sum_{j=1}^{N} \mathbf{A}_{ij}$ and $(\mathbf{D})_{ij} = 0$ for $i \neq j$,

The **combinatorial graph Laplacian** defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$,
and the **symmetric normalized Laplacian** $\mathcal{L} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$.

Signals on Graphs

Graph signal f in R^N, where |V|=N



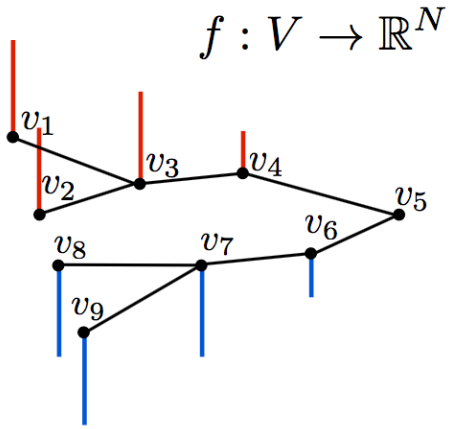
$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_9 \end{bmatrix} = \begin{bmatrix} 0.6 \\ 0.7 \\ 0.3 \\ \vdots \\ 0.7 \end{bmatrix}$$

Graph Laplacian $\mathcal{L} := \mathrm{D} - \mathrm{W}$, D: diagonal with sums of weights

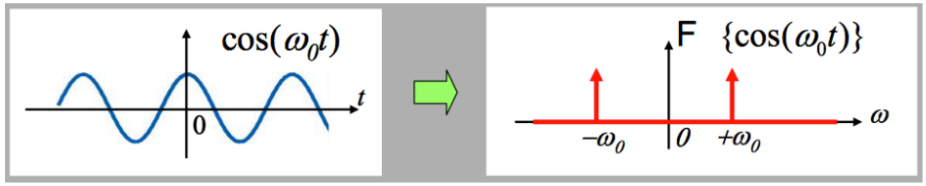
W: weight matrix

Normalized Graph Laplacian $\mathcal{L} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$

- The main approaches can be categorized into two families:
  - Vertex (spatial) domain designs
  - Frequency (graph spectral) domain designs



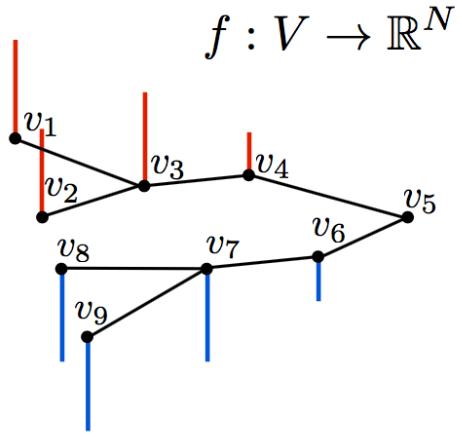
- Classical Fourier transform provides the frequency domain representation of the signals



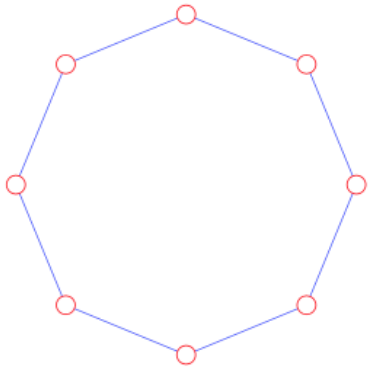
Source: http://www.physik.uni-kl.de

A notion of frequency for graph signals:

We need the graph Laplacian matrix



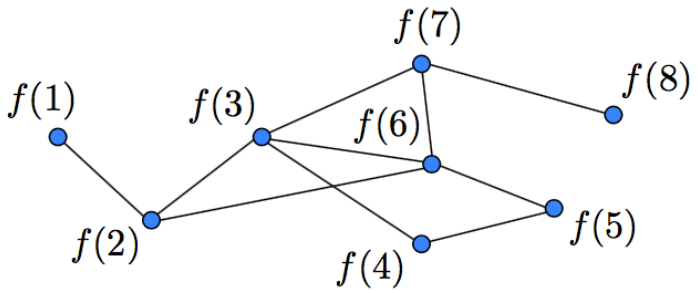
TRANSFORMATIONS ON GRAPHS



$$\mathbf{L} = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{pmatrix} - \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

- **A** and **D**: adjacency and degree matrices, $\mathbf{L} = \mathbf{D} - \mathbf{A}$: graph Laplacian
- **L** can be interpreted as a local (high-pass) operation on this graph
- Circulant matrix – Eigenvectors: DFT

Graph Laplacian



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

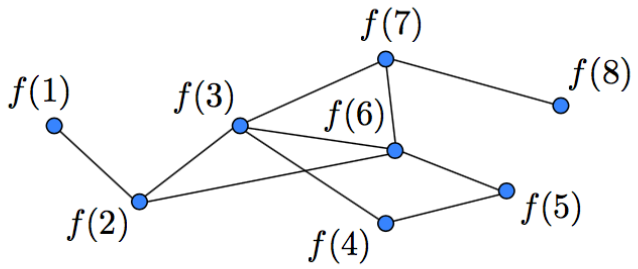
$$\begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & -1 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & -1 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix}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

L f

A difference operator:

$$Lf = \sum_{i,j=1}^{N} A_{ij} \left(f(i) - f(j) \right)$$

Graph Laplacian



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

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

L f

A difference operator:

$$Lf = \sum_{i,j=1}^{N} A_{ij} \left(f(i) - f(j) \right)$$

Laplacian quadratic form:

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

A measure of "smoothness" [Zhou04]

SPECTRAL PROPERTIES OF THE LAPLACIAN

- Denote by λ_i and $\mathbf{v}_i$ the eigenvalues and eigenvectors of $\mathbf{L}$
- $\mathbf{L}$ is positive semi-definite
  - $\Rightarrow \mathbf{x}^T\mathbf{L}\mathbf{x} = \frac{1}{2}\sum_{(i,j)\in\mathcal{E}} w_{ij}(x_i - x_j)^2 \geq 0$, for all $\mathbf{x}$
  - $\Rightarrow$ All eigenvalues are nonnegative, i.e. $\lambda_i \geq 0$ for all i
- A constant vector $\mathbf{1}$ is an eigenvector of $\mathbf{L}$ with eigenvalue 0

$$[\mathbf{L}\mathbf{1}]_i = \sum_{j\in\mathcal{N}(i)} w_{ij}(1-1) = 0$$

  - $\Rightarrow$ Thus, $\lambda_1 = 0$ and $\mathbf{v}_1 = (1/\sqrt{N})\,\mathbf{1}$
- In connected graphs, it holds that $\lambda_i > 0$ for $i = 2, \ldots, N$
  - $\Rightarrow$ Multiplicity$\{\lambda = 0\}$ = number of connected components

The eigenvalues of a graph characterize the topological structure of the graph

Examples :

(1) if $\lambda_1(G) = -\lambda_n(G)$, then G is bipartite;

(2) if $\lambda_2(G) = 0$, then G is complete multi-partite;

(3) if $\lambda_2(G) = -1$, then G is a complete graph;

(4) ...

In geographic studies, the eigenvalues and eigenvectors of a transportation network provide information about its connectedness. It is proven that the more highly connected in a transportation network G is, the larger is the largest eigenvalue $\lambda_1(G)$. [Tinkler, 72], [Roberts, 78].

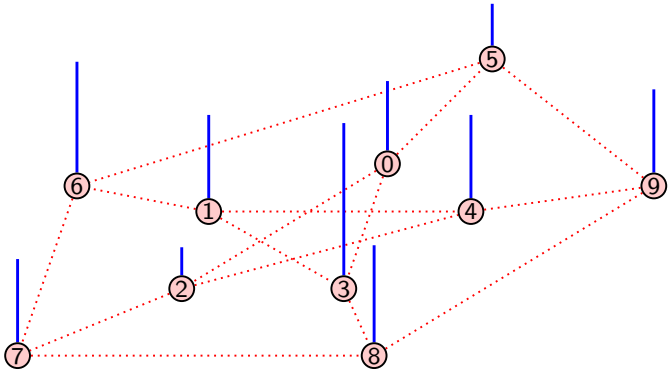
Given the numbers of vertices and edges, how to design a graph with larger $\lambda_1(G)$? – very interesting

GRAPH SIGNALS

- Consider graph $G = (\mathcal{V}, \mathcal{E}, W)$. Graph signals are mappings $x : \mathcal{V} \to \mathbb{R}$
  $\Rightarrow$ Defined on the vertices of the graph (data tied to nodes)

  Ex: Opinion profile, buffer congestion levels, neural activity, epidemic

- May be represented as a vector $\mathbf{x} \in \mathbb{R}^N$
  $\Rightarrow$ x_n denotes the signal value at the n-th vertex in $\mathcal{V}$
  $\Rightarrow$ Implicit ordering of vertices (same as in $\mathbf{A}$ or $\mathbf{L}$)



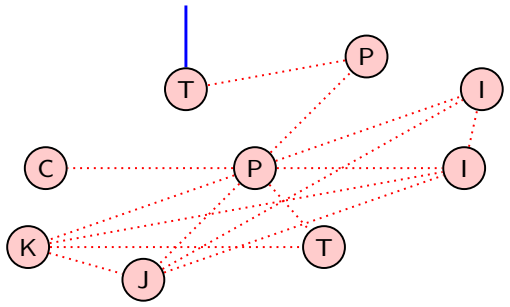
$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_9 \end{bmatrix} = \begin{bmatrix} 0.6 \\ 0.7 \\ 0.3 \\ \vdots \\ 0.7 \end{bmatrix}$$

- Data associated with links of G $\Rightarrow$ Use line graph of G

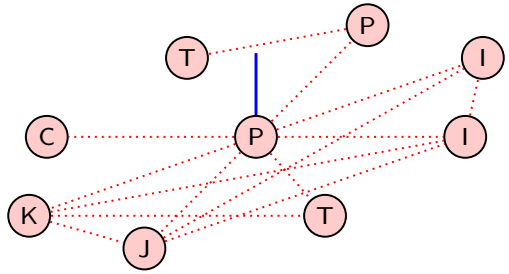
GRAPH EXAMPLE- GENE PROFILE

- Graphs representing gene-gene interactions
  - $\Rightarrow$ Each node denotes a single gene (loosely speaking)
  - $\Rightarrow$ Connected if their coded proteins participate in same metabolism
- Genetic profiles for each patient can be considered as a graph signal
  - $\Rightarrow$ Signal on each node is 1 if mutated and 0 otherwise

$$\mathbf{x}_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$



Sample patient 1 with subtype 1



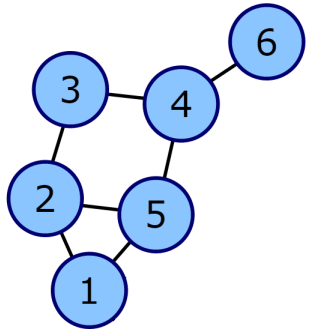
Sample patient 2 with subtype 1

$$\mathbf{x}_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

- To understand a graph signal, the structure of G must be considered

GRAPH-SHIFT OPERATOR

- To understand and analyze $\mathbf{x}$, useful to account for G's structure
- Associated with G is the graph-shift operator $\mathbf{S} \in \mathbb{R}^{N \times N}$
  $\Rightarrow S_{ij} = 0$ for $i \neq j$ and $(i, j) \notin \mathcal{E}$ (captures local structure in G)
- $\mathbf{S}$ can take nonzero values in the edges of G or in its diagonal

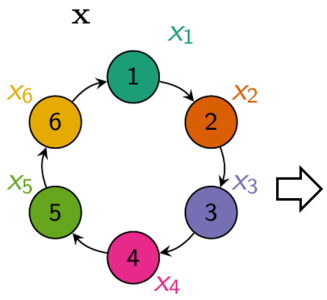


$$\mathbf{S} = \begin{pmatrix} S_{11} & S_{12} & 0 & 0 & S_{15} & 0 \\ S_{21} & S_{22} & S_{23} & 0 & S_{25} & 0 \\ 0 & S_{23} & S_{33} & S_{34} & 0 & 0 \\ 0 & 0 & S_{43} & S_{44} & S_{45} & S_{46} \\ S_{51} & S_{52} & 0 & S_{54} & S_{55} & 0 \\ 0 & 0 & 0 & S_{64} & 0 & S_{66} \end{pmatrix}$$

- Ex: Adjacency $\mathbf{A}$, degree $\mathbf{D}$, and Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$ matrices

RELEVANCE OF THE GRAPH SHIFT OPERATOR

- Q: Why is **S** called shift? A: Resemblance to time shifts



Directed Cycle

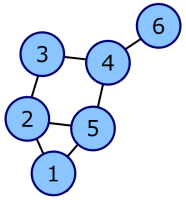
Set $\mathbf{S} = \mathbf{A_{dc}}$

$$\begin{pmatrix} 0 \\ 0 \\ x_2 \\ x_3 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ x_2 \\ x_3 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

LOCAL STRUCTURE OF THE GRAPH-SHIFT OPERATOR

S *represents a linear transformation that can be computed locally at the nodes of the graph. More rigorously, if* **y** *is defined as* $\mathbf{y} = \mathbf{S}\mathbf{x}$*, then node* i *can compute* y_i *if it has access to* x_j *at* $j \in \mathcal{N}(i)$*.*

- Straightforward because $[\mathbf{S}]_{ij} \neq 0$ only if $i = j$ or $(j, i) \in \mathcal{E}$



$$\Rightarrow \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{pmatrix} = \begin{pmatrix} S_{11} & S_{12} & 0 & 0 & S_{15} & 0 \\ S_{21} & S_{22} & S_{23} & 0 & S_{25} & 0 \\ 0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\ 0 & 0 & S_{43} & S_{44} & S_{45} & S_{46} \\ S_{51} & S_{52} & 0 & S_{54} & S_{55} & 0 \\ 0 & 0 & 0 & S_{64} & 0 & S_{66} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix}$$

- What if $\mathbf{y} = \mathbf{S}^2\mathbf{x}$?

  $\Rightarrow$ Like powers of **A**: neighborhoods

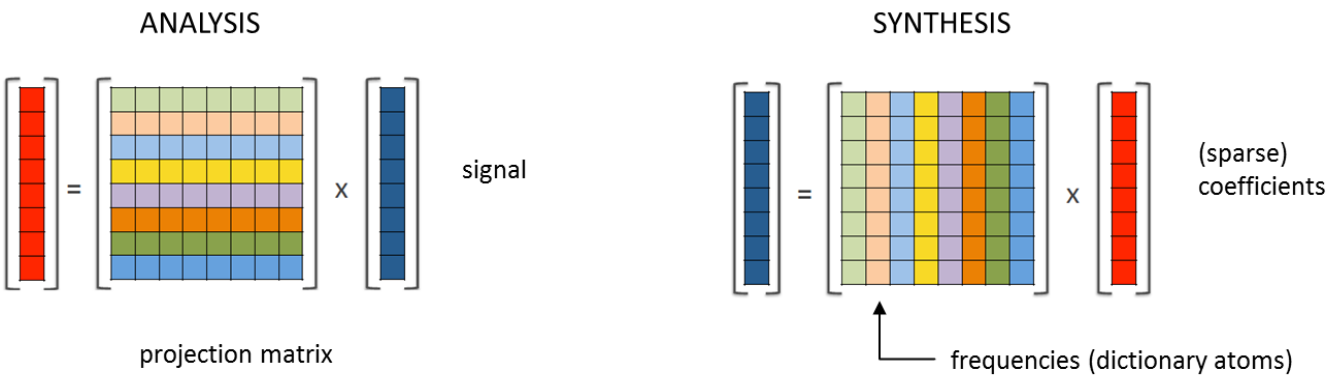
  $\Rightarrow$ y_i found using values within 2-hops

$$[\mathbf{S}^2]_{3,5} = S_{3,2}S_{2,5} + S_{3,4}S_{4,5}$$

$$\mathbf{S}^2 = \begin{pmatrix} S_{11} & S_{12} & 0 & 0 & S_{15} & 0 \\ S_{21} & S_{22} & S_{23} & 0 & S_{25} & 0 \\ 0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\ 0 & 0 & S_{43} & S_{44} & S_{45} & S_{46} \\ S_{51} & S_{52} & 0 & S_{54} & S_{55} & 0 \\ 0 & 0 & 0 & S_{64} & 0 & S_{66} \end{pmatrix} \begin{pmatrix} S_{11} & S_{12} & 0 & 0 & S_{15} & 0 \\ S_{21} & S_{22} & S_{23} & 0 & S_{25} & 0 \\ 0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\ 0 & 0 & S_{43} & S_{44} & S_{45} & S_{46} \\ S_{51} & S_{52} & 0 & S_{54} & S_{55} & 0 \\ 0 & 0 & 0 & S_{64} & 0 & S_{66} \end{pmatrix}$$

DFT ON GRAPHS

- Let $\mathbf{x}$ be a temporal signal, its DFT is $\tilde{\mathbf{x}} = \mathbf{F}^H \mathbf{x}$, with $F_{kn} = \frac{1}{\sqrt{N}} e^{+j\frac{2\pi}{N}kn}$
  - $\Rightarrow$ Equivalent description, provides insights
  - $\Rightarrow$ Oftentimes, more parsimonious (bandlimited)
  - $\Rightarrow$ Facilitates the design of SP algorithms: e.g., filters
- Many other transformations (orthogonal dictionaries) exist



- Q: What transformation is suitable for graph signals?

DFT ON GRAPHS

- Useful transformation? $\Rightarrow$ $\mathbf{S}$ involved in generation/description of $\mathbf{x}$
  $\Rightarrow$ Let $\mathbf{S} = \mathbf{V}\Lambda\mathbf{V}^{-1}$ be the shift associated with G
- The Graph Fourier Transform (GFT) of $\mathbf{x}$ is defined as

$$\tilde{\mathbf{x}} = \mathbf{V}^{-1}\mathbf{x}$$

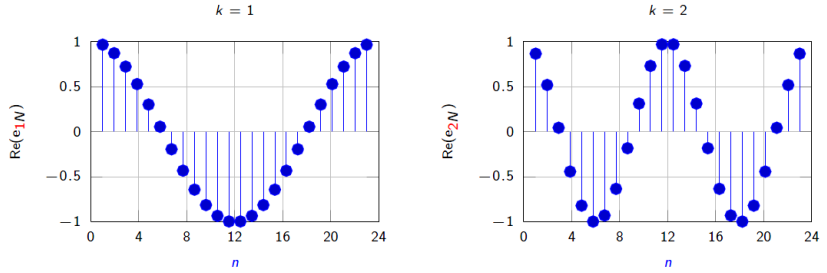
- While the inverse GFT (iGFT) of $\tilde{\mathbf{x}}$ is defined as

$$\mathbf{x} = \mathbf{V}\tilde{\mathbf{x}}$$

  $\Rightarrow$ Eigenvectors $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_N]$ are the frequency basis (atoms)
- Additional structure
  $\Rightarrow$ If $\mathbf{S}$ is normal, then $\mathbf{V}^{-1} = \mathbf{V}^H$ and $\tilde{x}_k = \mathbf{v}_k^H\mathbf{x} = < \mathbf{v}_k, \mathbf{x} >$
  $\Rightarrow$ Parseval holds, $\|\mathbf{x}\|^2 = \|\tilde{\mathbf{x}}\|^2$
- GFT $\Rightarrow$ Projection on eigenvector space of shift operator $\mathbf{S}$

EIGEN VALUES AS FREQUENCIES

- Columns of $\mathbf{V}$ are the frequency atoms: $\mathbf{x} = \sum_k \tilde{x}_k \mathbf{v}_k$
- Q: What about the eigenvalues $\lambda_k = \Lambda_{kk}$
  - $\Rightarrow$ When $\mathbf{S} = \mathbf{A}_{dc}$, we get $\lambda_k = e^{-j\frac{2\pi}{N}k}$
  - $\Rightarrow$ λ_k can be viewed as frequencies!!
- In time, well-defined relation between frequency and variation
  - $\Rightarrow$ Higher k $\Rightarrow$ higher oscillations
  - $\Rightarrow$ Bounds on total-variation: $TV(\mathbf{x}) = \sum_n (x_n - x_{n-1})^2$



- Q: Does this carry over for graph signals?
  - $\Rightarrow$ No in general, but if $\mathbf{S} = \mathbf{L}$ there are interpretations for λ_k
  - $\Rightarrow$ $\{\lambda_k\}_{k=1}^N$ will be very important when analyzing graph filters

- Designed in the vertex domain of the graph. Examples:
  - Graph wavelets [Crovella'03]
  - Approaches for WSN [Wang'06], [Wagner'05] [Shen-ICASSP08]

- 1-hop averaging transform

$$y[n] = \frac{1}{d_n} \sum_{m=1}^{N} A[n,m]x[m] \qquad \Rightarrow \qquad \mathbf{y} = \mathbf{D}^{-1}\mathbf{A}\mathbf{x} = \mathbf{P}_{rw}\mathbf{x}$$

- 1-hop difference transform

$$y[n] = \frac{1}{d_n} \sum_{m=1}^{N} A[n,m](x[n] - x[m]) \qquad \Rightarrow \quad \mathbf{y} = \mathcal{L}_{\mathbf{rw}}\mathbf{x} = \mathbf{x} - \mathbf{P}_{\mathbf{rw}}\mathbf{x}$$

- Graph Fourier Transform

$$\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T = \sum_{i=1}^{N} \lambda_i \mathbf{u}_i \mathbf{u}_i^T,$$

- GFT : projection onto the eigenvectors of the graph Laplacian

$$\hat{\mathbf{x}} = \mathbf{U}^T \mathbf{x}$$

- Inverse GFT:

$$\mathbf{x} = \mathbf{U}\,\hat{\mathbf{x}}$$

- The graph Laplacian eigenvectors associated with low frequencies vary slowly across the graph
- The eigenvectors associated with larger eigenvalues oscillate more rapidly

INTERPRETATION OF THE LAPLACIAN

- Consider a graph G, let $\mathbf{x}$ be a signal on G, and set $\mathbf{S} = \mathbf{L}$
  - $\Rightarrow \mathbf{y} = \mathbf{S}\mathbf{x}$ is now $\mathbf{y} = \mathbf{L}\mathbf{x} \Rightarrow y_i = \sum_{j \in \mathcal{N}(i)} w_{ij}(x_i - x_j)$
  - $\Rightarrow j$-th term is large if x_j is very different from neighboring x_i
  - $\Rightarrow y_i$ measures difference of x_i relative to its neighborhood
- We can also define the quadratic form $\mathbf{x}^T\mathbf{S}\mathbf{x}$

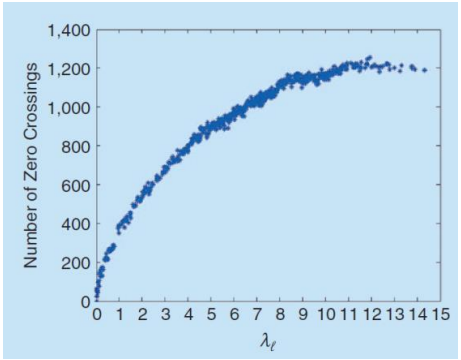
$$\mathbf{x}^T\mathbf{L}\mathbf{x} = \frac{1}{2}\sum_{(i,j)\in\mathcal{E}} w_{ij}(x_i - x_j)^2$$

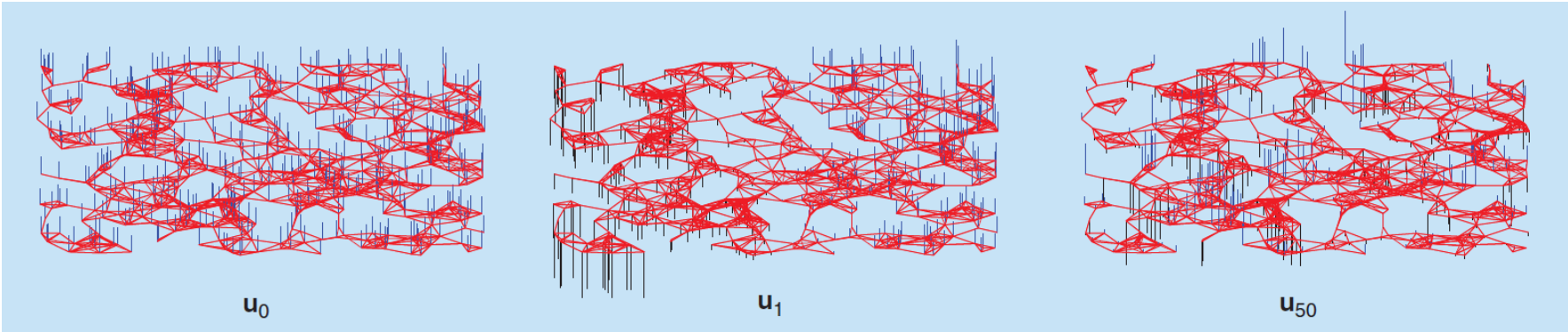
  - $\Rightarrow \mathbf{x}^T\mathbf{L}\mathbf{x}$ quantifies the (aggregated) local variation of signal $\mathbf{x}$
  - $\Rightarrow$ Natural measure of signal smoothness w.r.t. G
- Q: Interpretation of frequencies $\{\lambda_k\}_{k=1}^N$ when $\mathbf{S} = \mathbf{L}$?
  - $\Rightarrow$ If $\mathbf{x} = \mathbf{v}_k$, we get $\mathbf{x}^T\mathbf{L}\mathbf{x} = \lambda_k \Rightarrow$ local variation of $\mathbf{v}_k$
  - $\Rightarrow$ Frequencies account for local variation, they can be ordered
  - $\Rightarrow$ Eigenvector associated with eigenvalue 0 is constant

Graph Laplacian

Spectral properties $\mathcal{L}\mathbf{u}_\ell = \lambda_\ell \mathbf{u}_\ell$

- Laplacian is Positive Semi-definite matrix
- Eigenvalues: $0=\lambda_1(L) \leq \lambda_2(L) \leq \ldots \leq \lambda_{N-1}(L)$
- Eigen-pair system $\{\lambda_k, \mathbf{u}_k\}$ provides Fourier-like interpretation (GFT)





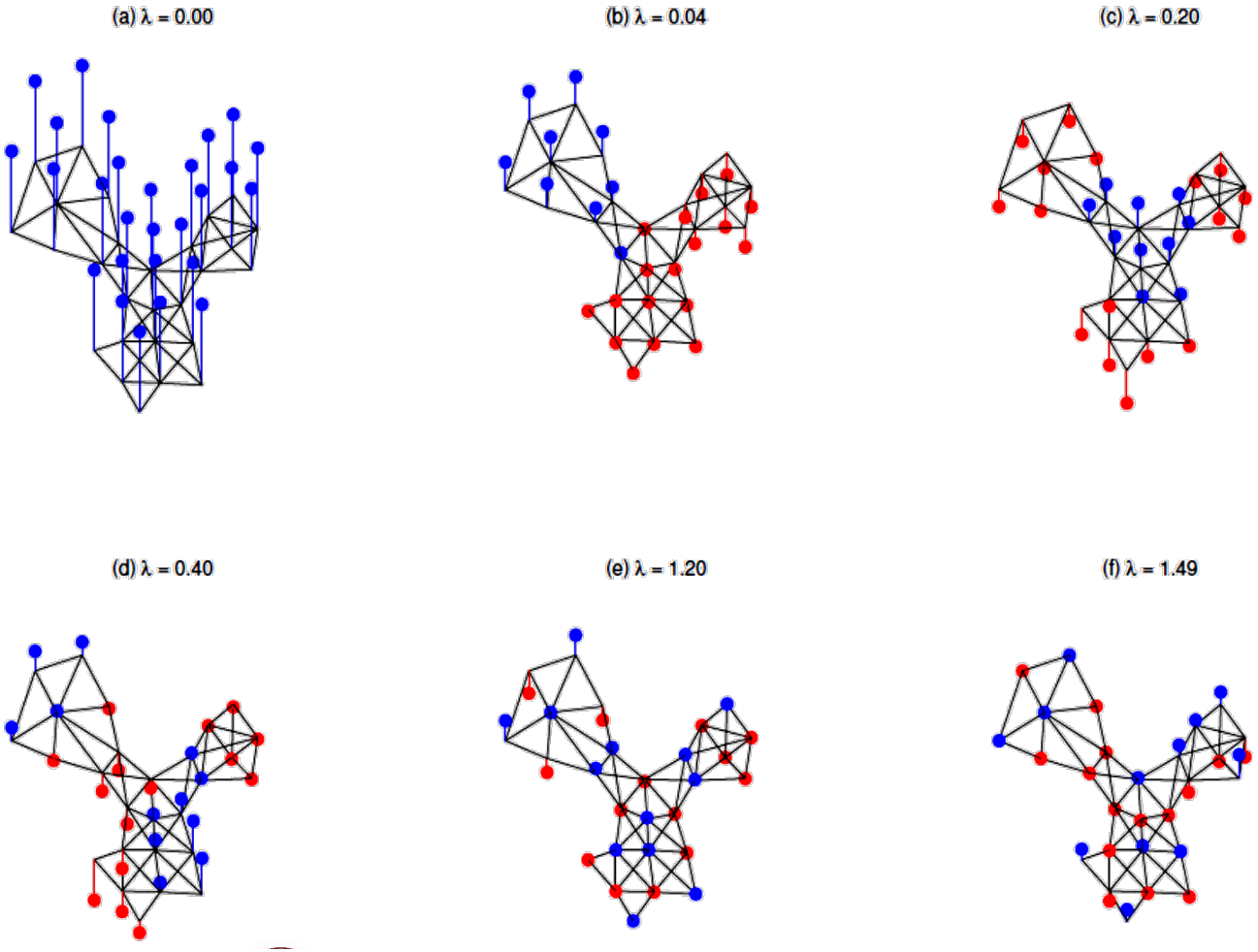
$L = \chi \Lambda \chi^T$

Low frequency → High frequency

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

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

Eigenvectors of Graph Laplacian



Importance

For connected graphs, the Laplacian eigenvector $\mathbf{u}_0$ associated with the eigenvalue 0 is constant and equal to $\frac{1}{\sqrt{N}}$ at each vertex.

The graph Laplacian eigenvectors associated with low frequencies vary slowly across the graph.

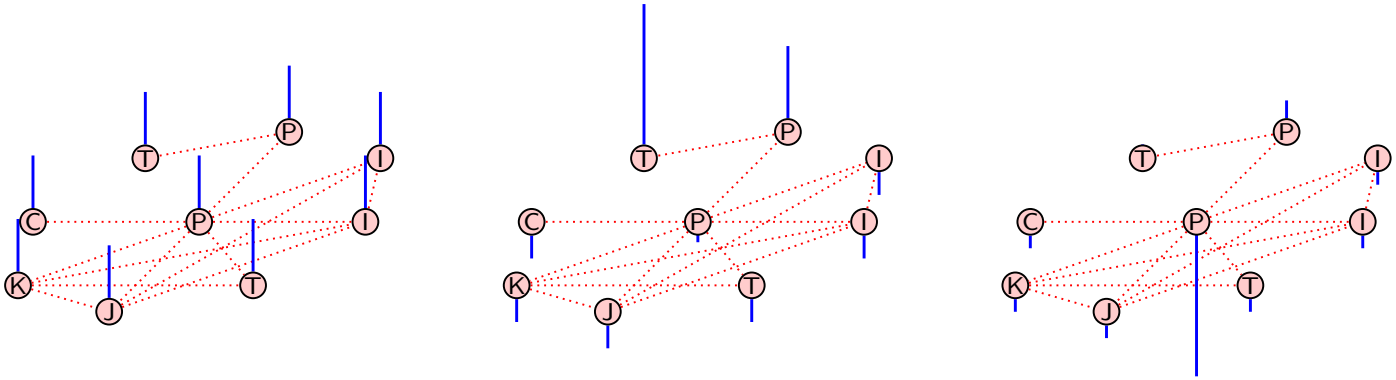
If two vertices are connected by an edge with a large weight, the values of the eigenvector at those locations are similar. The eigenvectors associated with larger eigenvalues oscillate more rapidly and are more likely to have dissimilar values on vertices connected by an edge with high weight.

RECALL

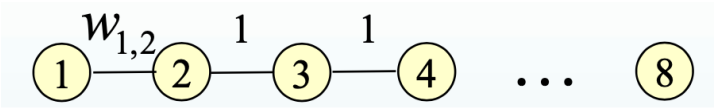
- Graph Laplacian Matrix $\mathbf{L} = \mathbf{D} - \mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}'$
- Eigen-vectors of $\mathbf{L}$: $\mathbf{U} = \{\mathbf{u}_k\}_{k=1:N}$
- Eigen-values of $\mathbf{L}$: $diag\{\mathbf{\Lambda}\} = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_N$
- **Eigen-pair system $\{(\lambda_k, \mathbf{u}_k)\}$ provides Fourier-like interpretation — Graph Fourier Transform (GFT)**

FREQUENCIES OF THE LAPLACIAN

- Laplacian eigenvalue λ_k accounts for the local variation of $\mathbf{v}_k$
  $\Rightarrow$ Let us plot some of the eigenvectors of $\mathbf{L}$ (also graph signals)
- Ex: gene network, $N = 10$, $k = 1$, $k = 2$, $k = 9$



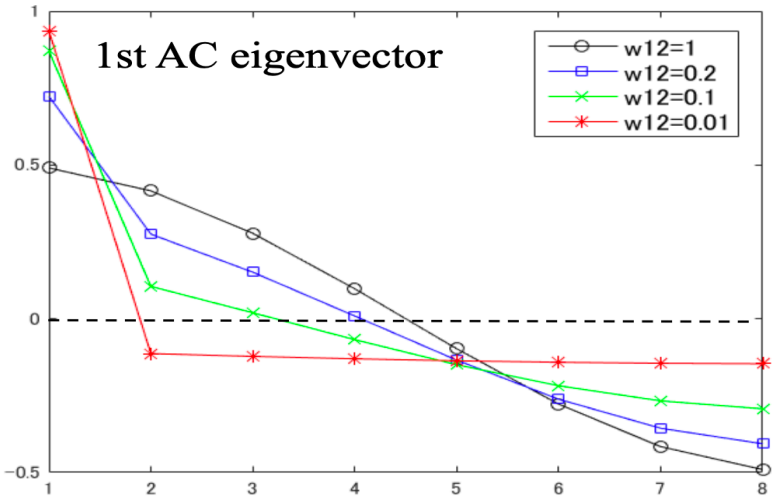
Graph Fourier Transform (GFT) is eigen-matrix of graph Laplacian L.



$$L u_i = \lambda_i u_i$$

eigenvalue

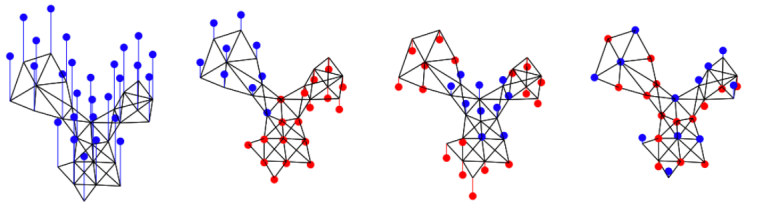
eigenvector



- Edge weights affect shapes of eigenvectors.
- Eigenvalues (≥ 0) as graph frequencies.
- Constant eigenvector is DC.
- # zero-crossings increases as λ increases.

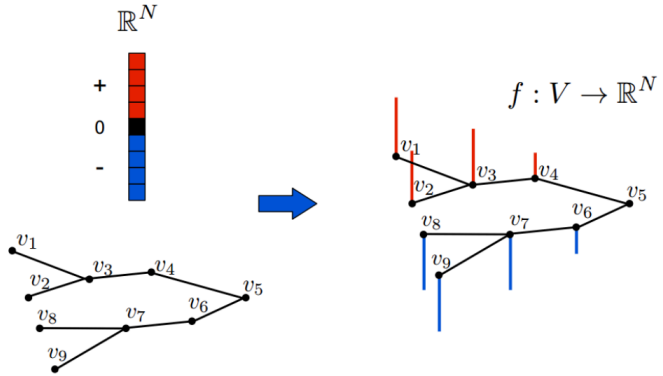
$$L = \begin{bmatrix} | & & | \\ \chi_0 & \cdots & \chi_{N-1} \\ | & & | \end{bmatrix} \begin{bmatrix} \lambda_0 & & 0 \\ & \ddots & \\ 0 & & \lambda_{N-1} \end{bmatrix} \begin{bmatrix} — & \chi_0 & — \\ & \cdots & \\ — & \chi_{N-1} & — \end{bmatrix}$$

χ Λ χ^T



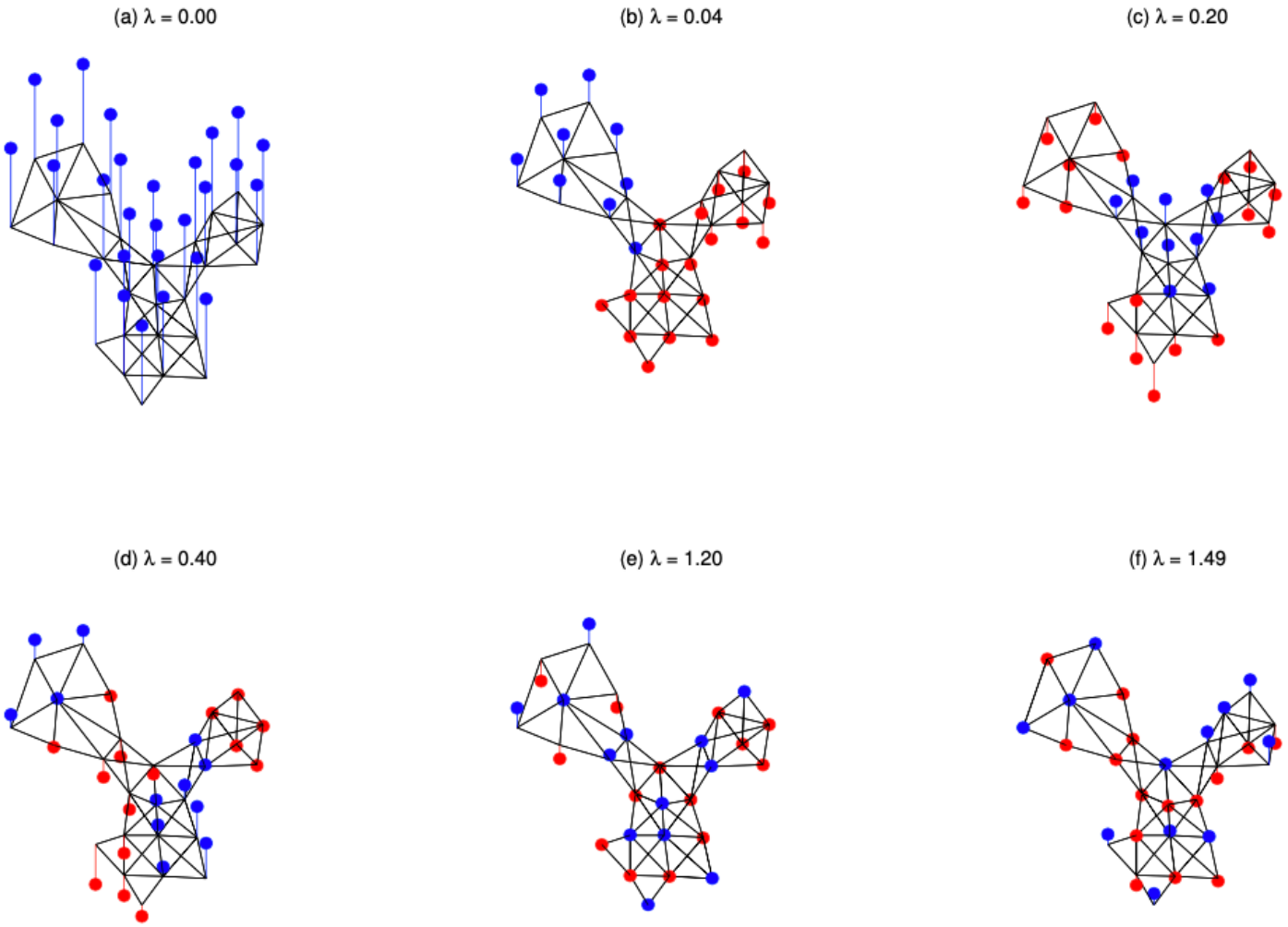
χ_0 $\lambda_0 = 0.00$ χ_1 $\lambda_1 = 0.04$ χ_2 $\lambda_2 = 0.20$ χ_3 $\lambda_3 = 1.49$

Visual example of graph frequencies (Ortega and Dong slides)

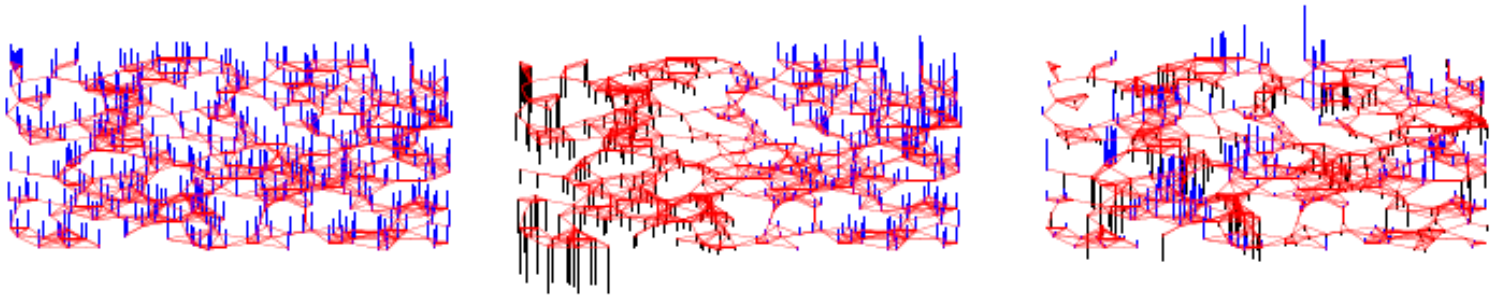


Simple graph signal example

Eigenvectors of graph Laplacian



Example



$\mathbf{u}_0$ $\mathbf{u}_1$ $\mathbf{u}_{50}$

Fig. 2. Three graph Laplacian eigenvectors of a random sensor network graph. The signals' component values are represented by the blue (positive) and black (negative) bars coming out of the vertices. Note that $\mathbf{u}_{50}$ contains many more zero crossings than the constant eigenvector $\mathbf{u}_0$ and the smooth *Fiedler vector* $\mathbf{u}_1$.

EXAMPLE

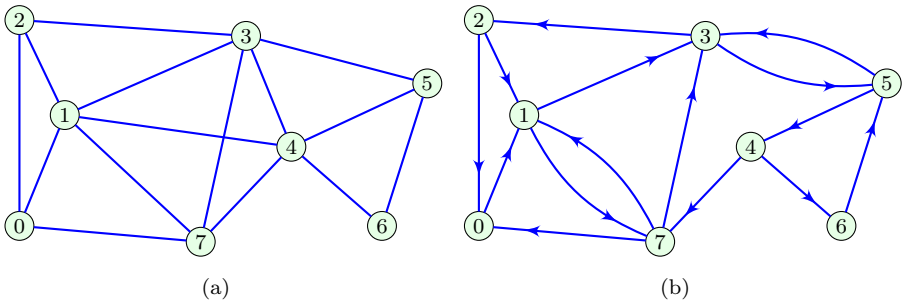


Fig. 2 Examples of: (a) Undirected graph and (b) Directed graph.

The adjacency matrices for the graphs from Fig. 2(a) and (b) are

$$\mathbf{A} = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix} \end{array}, \quad \mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad (1)$$

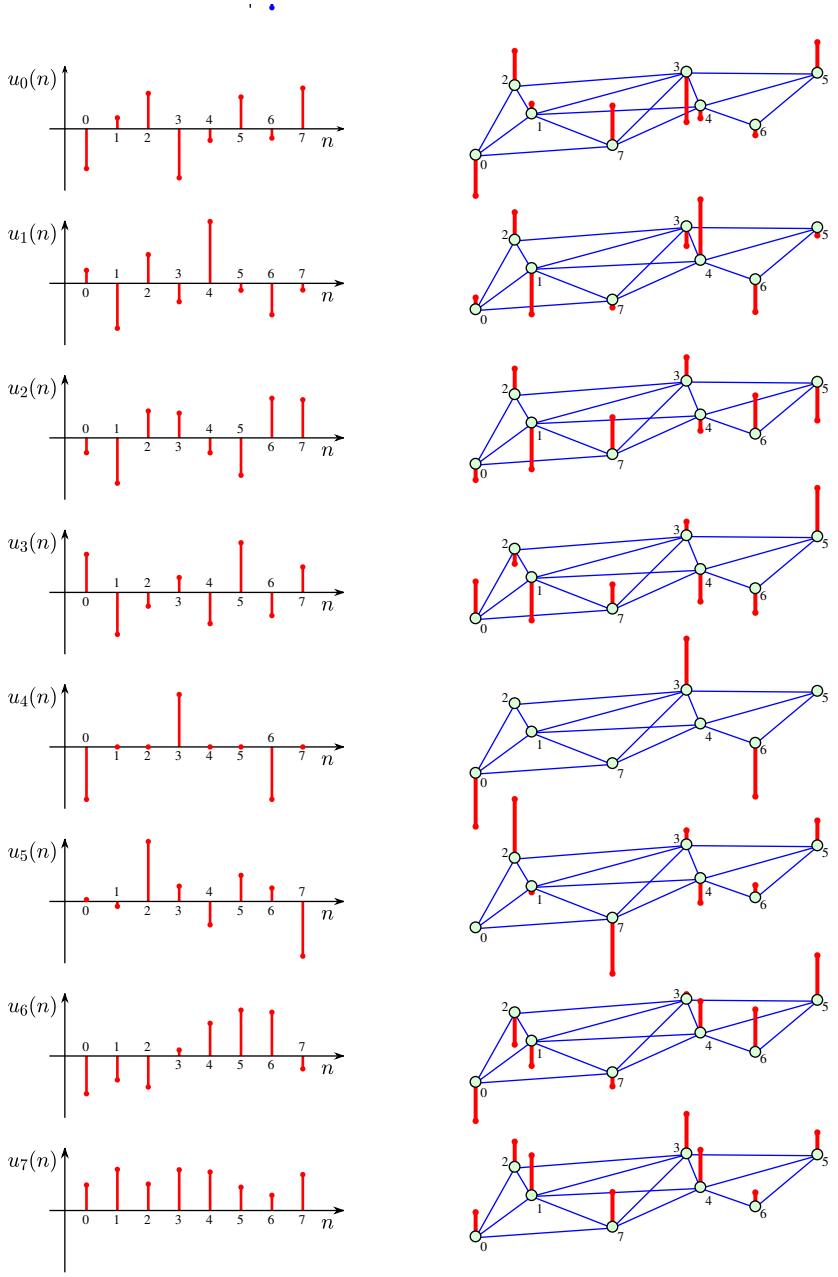
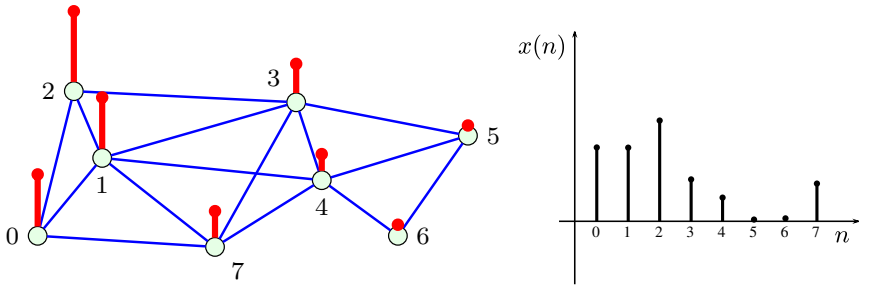
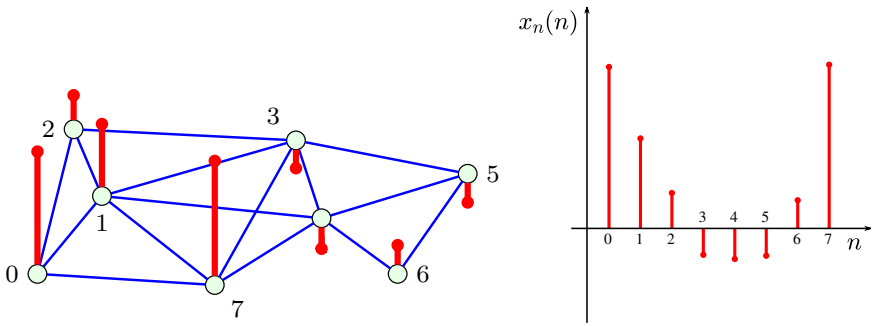


Fig. 10 Eigenvalues λ_k and corresponding eigenvectors $u_k(n)$ for the adjacency matrix of the graph presented in Fig. 2(a). The eigenvectors are shown on the vertex index line (left) and on the graph (right).

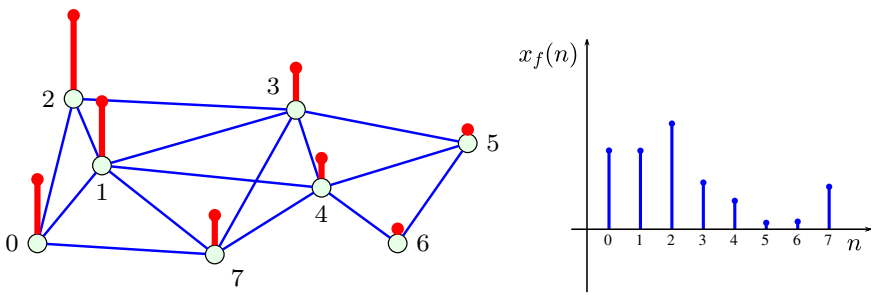
FILTERING of the Graph Signal



(a) original signal



(b) noisy signal



(c) filtered signal

Fig. 26 Signal filtering example. Original signal (a), noisy signal (b) and filtered signal (c). Low pass filtering with two largest eigenvalues is applied.

FILTERING ON GRAPH SIGNALS

Ljubiša Stanković, Danilo P. Mandic, Miloš Daković,
Ilya Kisil, Ervin Sejdić, and Anthony G. Constantinides

Understanding the Basis of Graph Signal Processing via an Intuitive Example-Driven Approach

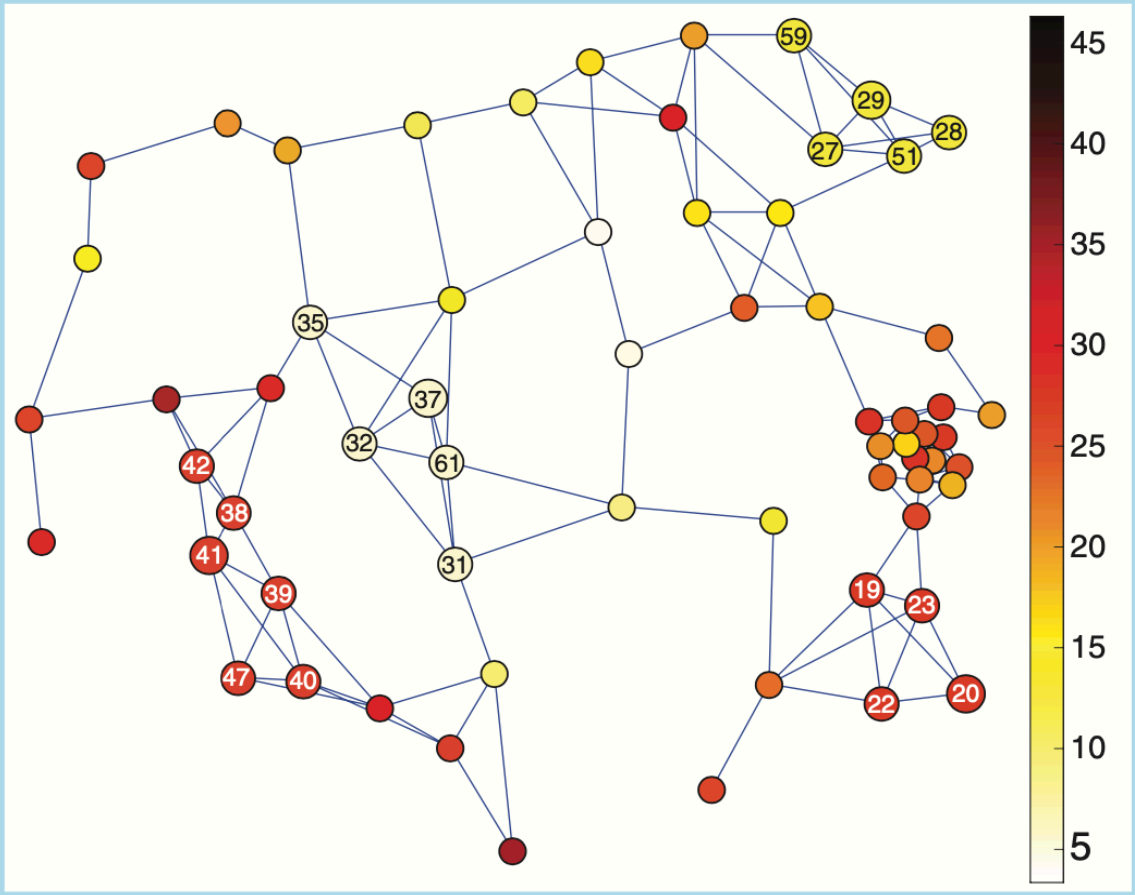
Smoothness and Filtering on a Graph

The quadratic form of a graph signal on an undirected graph is given by

$$E_x = \mathbf{x}^T\mathbf{L}\mathbf{x} = \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} W_{nm}(x(n)-x(m))^2$$

and can be used to define signal smoothness, since small values of the squared local deviation, $(x(n)-x(m))^2$, correspond to a smooth, slow-varying signal. For a constant signal, $\mathbf{x}=\mathbf{c}$, we, therefore, have $E_x = 0$.

Physically, the minimum of $\mathbf{x}^T\mathbf{L}\mathbf{x}$ implies the smoothest possible signal, and to arrive at this solution we may employ steepest descent. Then, the signal value at an iteration p is adjusted in the opposite direction of the gradient, toward the minimum of $\mathbf{x}^T\mathbf{L}\mathbf{x}$. The gradient of this quadratic form is $\partial E_x/\partial \mathbf{x}^T = 2\mathbf{L}\mathbf{x}$, which yields the iterative procedure



(a)

$$\mathbf{x}_{p+1} = \mathbf{x}_p - \alpha \mathbf{L}\mathbf{x}_p = (\mathbf{I} - \alpha \mathbf{L})\mathbf{x}_p.$$

Notice that the signal $\mathbf{x}_{p+1}$ can be considered as an output of the first-order system in (11), with $h_1 = -\alpha$, and this relation can be used for simple and efficient filtering of graph signals.

Since the minimum of the quadratic form $\mathbf{x}^T\mathbf{L}\mathbf{x}$ corresponds to a constant signal, to avoid obtaining only a constant steady state (that is, to also account for the slow-varying part of the graph signal), the aforementioned iteration process can be used in alternation with $\mathbf{x}_{p+2} = (\mathbf{I} + \beta \mathbf{L})\mathbf{x}_{p+1}$. A compact form of these two iterative processes is known as Taubin's $\alpha - \beta$ algorithm and is given by

$$\mathbf{x}_{p+2} = (\mathbf{I} + \beta \mathbf{L})(\mathbf{I} - \alpha \mathbf{L})\mathbf{x}_p. \qquad \text{(S1)}$$

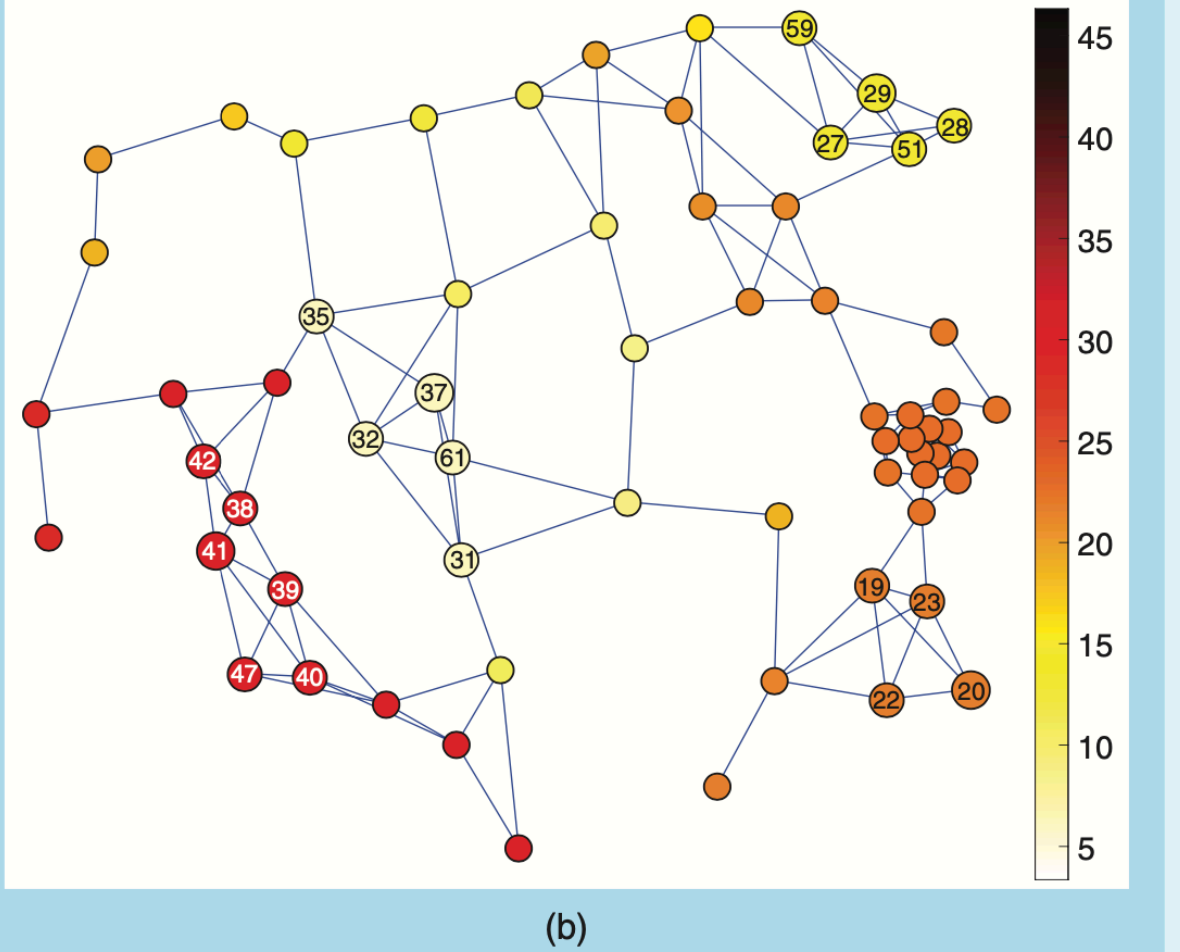


FIGURE S1. An illustration of low-pass filtering on a graph. (a) The original noisy signal. (b) The filtered signal. The graph signal intensity is designated by the vertex color.

For appropriate values of α and β, this system can give a good and very simple approximation of a low-pass graph filter with transfer function $H(\lambda_k) = (1 + (\beta - \alpha)\lambda_k - \alpha\beta\lambda_k^2)^P$, and in P iterations, where k denotes the spectral index (see the section "Spectral Domain Graph Filter Design").

In our experiment, the original noisy signal from Figure 3 was filtered using Taubin's algorithm, with $\alpha = 0.2$ and $\beta = 0.1$. After 50 iterations, the signal-to-noise ratio improved from the original $\text{SNR}_0 = 14.2$ dB to 26.8 dB (see Figure S1). With these parameters, the transfer function, $H(\lambda_k)$, retained seven out of 64 spectral components in the signal (with an attenuation lower than 3 dB).

Vertex Clustering

The term *vertex clustering* here refers to the task of identifying and arranging the vertices of a graph into nonoverlapping vertex subsets, with data in each subset expected to exhibit relative similarity in some sense. One efficient approach to vertex clustering is based on spectral graph analysis. For a graph with N vertices, the orthogonal eigenvectors of its Laplacian build an N-dimensional space, called the *spectral space*. The elements $u_k(n)$ of the eigenvector $\mathbf{u}_k$, $k = 1, 2, \ldots, N$, can then be assigned to vertices n, $n = 1, 2, \ldots, N$ to form an N-dimensional spectral vector $\mathbf{q}_n = [u_1(n), u_2(n), \ldots, u_N(n)]$. The elements of the first eigenvector, $\mathbf{u}_1$, of the graph Laplacian are constant and are omitted, since they do not convey any spectral difference to the graph vertices.

For the purpose of vertex clustering, the original N-dimensional spectral vector space may be reduced to a new L-dimensional spectral space ($L < N$), where the spectral vectors,

$$\mathbf{q}_n = [u_2(n), u_3(n), \ldots, u_{L+1}(n)],$$

are used to define spectral similarity between any two vertices, n and m, as $\|\mathbf{q}_n - \mathbf{q}_m\|_2$. Vertex clustering is then performed by grouping spectrally similar vertices.

The simplest (and most widely used) case is when only one eigenvector, $\mathbf{u}_2$, is used for spectral clustering, whereby the order of vertices in the sorted $\mathbf{u}_2$ corresponds to its smoothest representation. This procedure can be used for ordering the vertices in graphs, even if we desire to perform any form of classical presentation or processing with vertices on a path graph, as in Figure 1(c).

The spectral vector, $\mathbf{q}_n$, can be either used to designate a position of a vertex in a new low L-dimensional space, or it can be used for coloring of the vertices at their original positions. For the graph from Figure 2, such coloring was performed using the spectral vector elements $\mathbf{q}_n = [u_2(n), u_3(n), u_4(n)]$ as color coordinates for the vertex n (see Figure S2). Similar colors indicate high spectral similarity.

Note that vertex clustering is a signal-independent operation. It just roughly indicates the expected relation between sensor data values on the considered graph and suggests that data processing operations (including processing of the signal from Figure 3) will be predominantly localized within these clusters.

Formally, the so-achieved reduction in spectral vertex dimensionality, from the original N eigenvectors to L eigenvectors with lowest variations (with the smallest smoothness index $\mathbf{u}_k^T \mathbf{L} \mathbf{u}_k = \lambda_k$), corresponds to low-pass filtering in graph signal processing, whereby a signal with N spectral components is projected onto a reduced spectral space with L slowest-varying spectral components, within a given set of basis functions (cf. truncated Fourier representation).

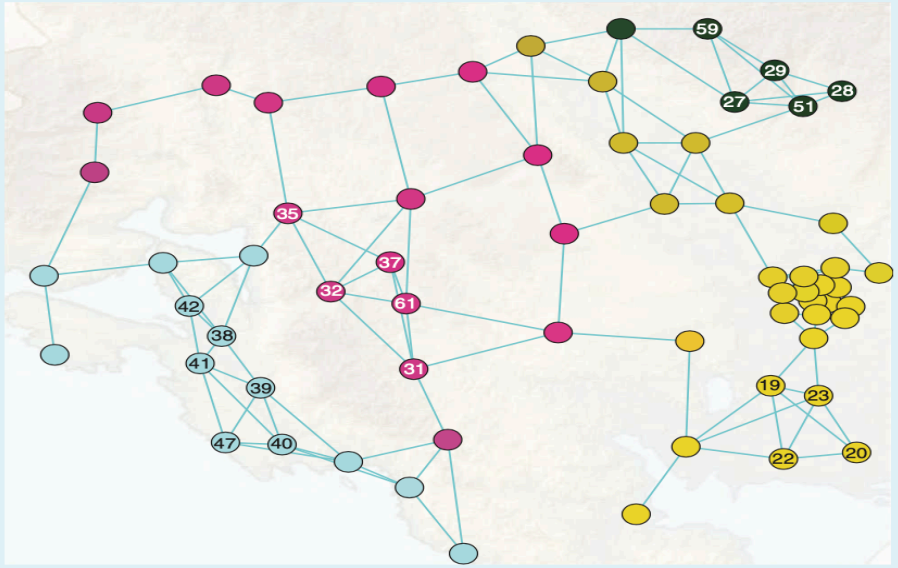
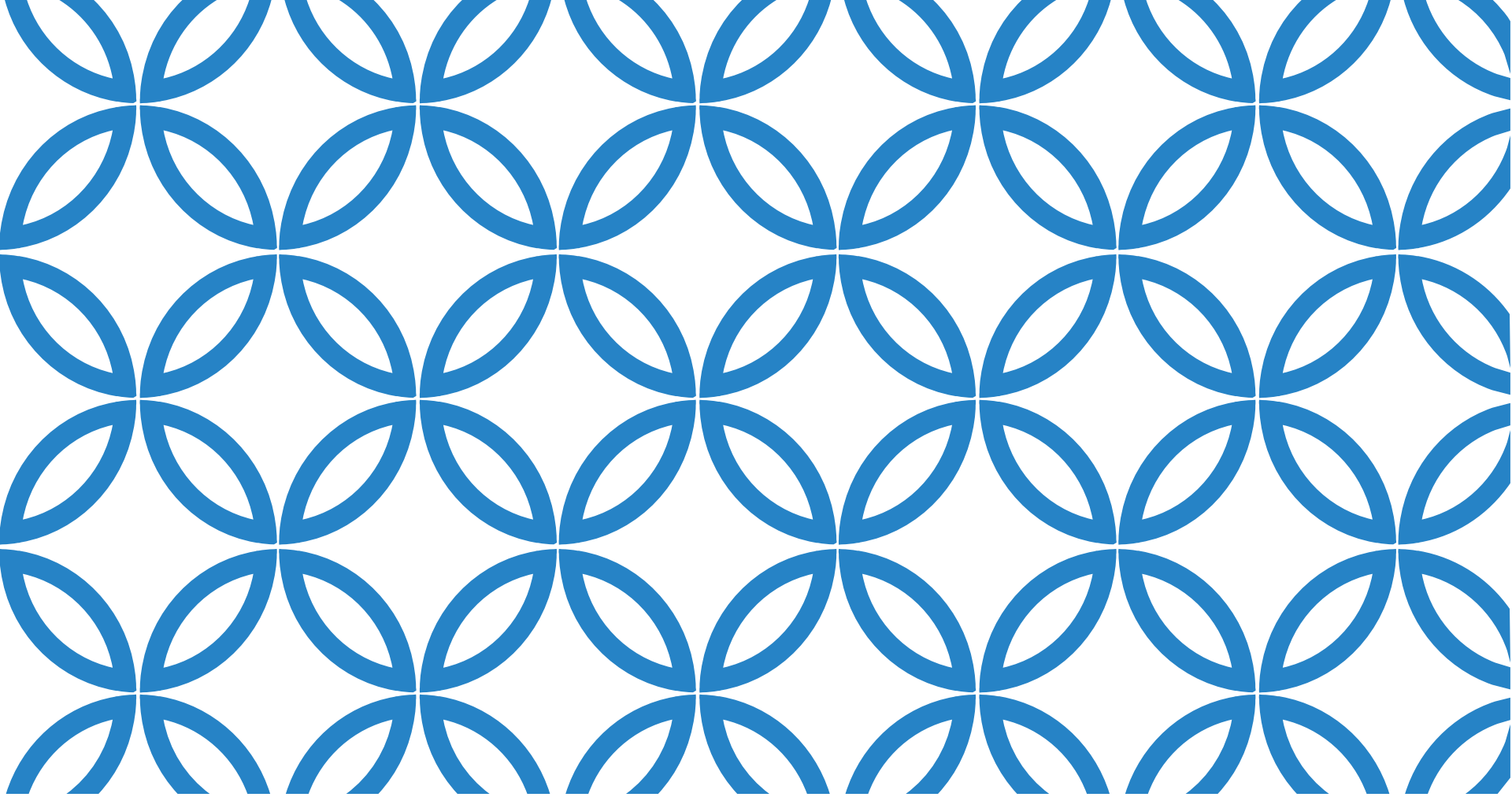


FIGURE S2. The vertices in the graph from Figure 2 have been colored using the spectral vectors $\mathbf{q}_n = [u_2(n), u_3(n), u_4(n)]$ as red, green, blue color coordinates.



SAMPLING SMOOTH SIGNALS

REFERENCES

A Distributed Tracking Algorithm for Reconstruction of Graph Signals

February 2015 · IEEE Journal of Selected Topics in Signal Processing 9(4)

DOI: 10.1109/JSTSP.2015.2403799

Source · arXiv

Xiaohan Wang · Mengdi Wang · Yuantao Gu

Exploiting Structure In Data: Sampling and Signal Processing on Graphs

Submitted in partial fulfillment of the requirements for
the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Rohan Anilkumar Varma

B.S Electrical Engineering and Computer Sciences, University of California, Berkeley
B.A Economics, University of California, Berkeley
B.A Statistics, University of California, Berkeley
M.S Electrical and Computer Engineering, Carnegie Mellon University

SMOOTH SIGNALS

- Sampling on graph signals algorithms can be considered as an extension of the Nyquist sampling to irregular signal domains- such sensor networks data.
- We learned how Graphs present a natural way to represent WSN measurements. Each graph vertex may represent one data point associated with one sensor.
- The graph edges can signify distances between sensory nodes or other pairwise relationships in some feature space.
- It is often that the measurements tend to be smoothly varying among sensors that are located in proximity.
- Thus, we seek to develop ways by which we exploit this characteristic of the network data to perform efficient (w.r.t transmission, storage, etc.) sensory data sampling. Like what we did using CS.
- This, ultimately, necessitates the development of signal reconstruction tools for recovering such signals.

SAMPLING SMOOTH SIGNALS

- Sampling of bandlimited graph signals that is analogous to down-sampling discrete-time signals such that we can recover the signal perfectly.
- It is possible to sample efficiently on graphs by using a structured sampling procedures and as result gain computational efficiency.
- Minimax lower bounds can be derived for sampling smooth graph signals.
- This will guide in analysing and designing optimal sampling and reconstruction algorithms
- This presents tools for recovering smooth graph signals from noisy or corrupted measurements by formulating an optimization problem that minimizes the variation of the signal over the graph.
- Furthermore, it provides a framework for recovering smooth signals on product graphs by exploiting the low rank structure of these signals.

SAMPLING OF BANDLIMITED GRAPH SIGNALS

3.1 Sampling Theory for Bandlimited Graph Signals

In this section, we consider the classical signal processing task of sampling theory within the framework of DSP_G. Sampling theory is a key topic in signal processing[38,39]. As the bridge connecting sequences and functions, classical sampling theory shows that a bandlimited function can be perfectly recovered from its sampled sequence if the sampling rate is high enough[40]. More generally, we can treat any decrease in dimension via a linear operator as sampling, and, conversely, any increase in dimension via a linear operator as interpolation[38,41]. Formulating a sampling theory in this context is equivalent to moving between higher- and lower-dimensional spaces while ensuring perfect recovery.

A sampling theory for graphs has interesting implications and applications. For example, given a graph representing friendship connectivity in Facebook, we can just sample a small fraction of users and query their hobbies. We then can recover all users' hobbies. The task of sampling on graphs is, however, not well understood[42,10]. It is challenging because graph signals lie on complex, irregular structure, where many classical concepts are ill-posed, such as downsampling[43]. It is even more challenging to find a graph structure that is associated with the sampled signal coefficients. For example, in the Facebook example, we sample a small fraction of users. An associated

SAMPLING OF BANDLIMITED GRAPH SIGNALS

graph structure would allow us to infer new connectivity between those sampled users, even when they are not directly connected in the original graph.

Some previous work on sampling theory[9,42] considers graph signals that are uniquely sampled onto some given subset of nodes. This approach is not consistent with classical sampling theory and applies to undirected graphs only. It also does not explain how a graph structure supports these sampled coefficients.

The assumption that graph signals vary slowly or are smooth over the graph is a natural one to make. Many real world graph signals like sensor network data and biological network data are smooth, or exhibit bandlimited behavior, or have known limited support with respect to the graph Fourier transform. For example, in the context of semi-supervised classification on graphs, each vertex represents one data point to which a label is associated and a graph can be formed by connecting vertices with weights corresponding to the affinity or distance between the data points in some feature space. It is then natural to assume that the *label signal* has slow variation or is *smooth* on the graph and consequently approximately bandlimited. Since labeled instances are rare or expensive to collect, devising efficient yet frugal sampling algorithms on large complex graphs is of significant interest. Here we propose a sampling theory for signals that are supported on either directed or undirected graphs. Perfect recovery is possible for graph signals bandlim-

SAMPLING OF BANDLIMITED GRAPH SIGNALS

a new graph signal whose corresponding graph structure is constructed from the original graph structure. The proposed sampling theory follows Chapter 5 from[38] and is consistent with classical sampling theory. We further establish the connection to the theories of frames with maximal robustness to erasures and compressed sensing, show a principle to choose the optimal sampling operator, and show how random sampling works on circulant graphs and Erdős-Rényi graphs. To handle full-band graphs signals, we propose graph filter banks to force graphs signals to be bandlimited. Finally, we validate the proposed sampling theory on three simulated datasets of Erdős-Rényi graphs, small-world graphs, scale-free graphs, and a real-world dataset of online blogs. We show that for each case, the proposed sampling theory achieves perfect recovery with high probabilities.

SAMPLING OF BANDLIMITED GRAPH SIGNALS: NOTATION

In general, $\mathbf{V}$ may not be orthonormal; to restrict its behavior, we assume that

$$\alpha_1 \|\mathbf{x}\|_2^2 \leq \|\mathbf{V}\,\mathbf{x}\|^2 \leq \alpha_2 \|\mathbf{x}\|_2^2, \quad \text{for all } \mathbf{x} \in \mathbb{R}^N,$$

where $\alpha_1, \alpha_2 > 0$, that is, $\mathbf{V}$ is a Riesz basis with stability constants α_1, α_2[38]. The eigenvalues $\lambda_0, \ldots \lambda_{N-1}$ of $\mathbf{A}$, represent frequencies on the graph[18].

Symbol	Description	Dimension
$\mathbf{A}$	graph shift	$N \times N$
$\mathbf{V}$	inverse graph Fourier transform matrix	$N \times N$
Ψ	sampling operator	$M \times N$
Φ	interpolation operator	$N \times M$
$\mathbf{x}$	graph signal	N
$\widehat{\mathbf{x}}$	graph signal in the frequency domain	N
$\mathcal{M}$	sampled indices	
$\mathbf{x}_{\mathcal{M}}$	sampled signal coeffcients of $\mathbf{x}$	M
$\widehat{\mathbf{x}}_{(K)}$	first K coeffcients of $\widehat{\mathbf{x}}$	K
$\widehat{\mathbf{x}}_{(-K)}$	except first K coeffcients of $\widehat{\mathbf{x}}$	K
$\mathbf{V}_{(K)}$	first K columns of $\mathbf{V}$	$N \times K$
$\mathbf{V}_{(-K)}$	except first K columns of $\mathbf{V}$	$N \times (N-K)$
$\mathbf{V}^{-1}_{(K)}$	first K rows of $\mathbf{V}^{-1}$	$K \times N$
$\mathbf{V}^{-1}_{(-K)}$	except first K rows of $\mathbf{V}^{-1}$	$(N-K) \times N$

Table 3.1: Key notation used in this chapter

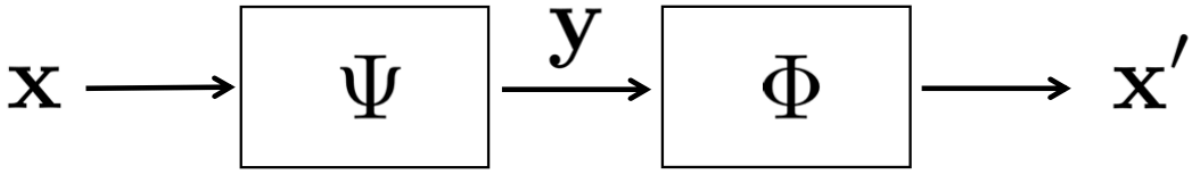
3.1.1 Sampling on Graphs

In this section, we propose a sampling theory for graph signals. We show that perfect recovery is possible for graph signals bandlimited under the graph Fourier transform, and a new graph shift for the sampled signal coefficients is constructed from the original graph shift. A toy example is shown to illustrate the proposed sampling theory. We further analyze the proposed sampling theory by showing the relations to previous theories, a principle to choose the optimal sampling operator, how random sampling works, and a graph filter bank to handle full-band graph signals.

Suppose that we want to sample M coefficients in a graph signal $\mathbf{x} \in \mathbb{R}^N$ to produce a sampled part $\mathbf{x}_{\mathcal{M}} \in \mathbb{R}^M$ $(M < N)$, where $\mathcal{M}$ denotes the sequence of *sampled* indices, $\mathcal{M} \subset \{0, 1, \cdots, N-1\}$ and $|\mathcal{M}| = M$. We then interpolate $\mathbf{x}_{\mathcal{M}}$ to get $\mathbf{x}' \in \mathbb{R}^N$, which recovers $\mathbf{x}$ either exactly or approximately. The sampling operator Ψ is a linear mapping from $\mathbb{R}^N$ to $\mathbb{R}^M$, defined as

$$\Psi_{i,j} = \begin{cases} 1, & j = \mathcal{M}_i; \\ 0, & \text{otherwise}, \end{cases} \tag{3.2}$$

and the interpolation operator Φ is a linear mapping from $\mathbb{R}^M$ to $\mathbb{R}^N$ (see Figure 3.1),



$$\text{sampling :} \qquad \mathbf{x}_{\mathcal{M}} = \Psi\mathbf{x} \in \mathbb{R}^M,$$

$$\text{interpolation :} \qquad \mathbf{x}' = \Phi\mathbf{x}_{\mathcal{M}} = \Phi\Psi\mathbf{x} \in \mathbb{R}^N.$$

Perfect recovery happens for all $\mathbf{x}$ when $\Phi\Psi$ is the identity matrix. This is not possible in general because $\text{rank}(\Phi\Psi) \leq M < N$.

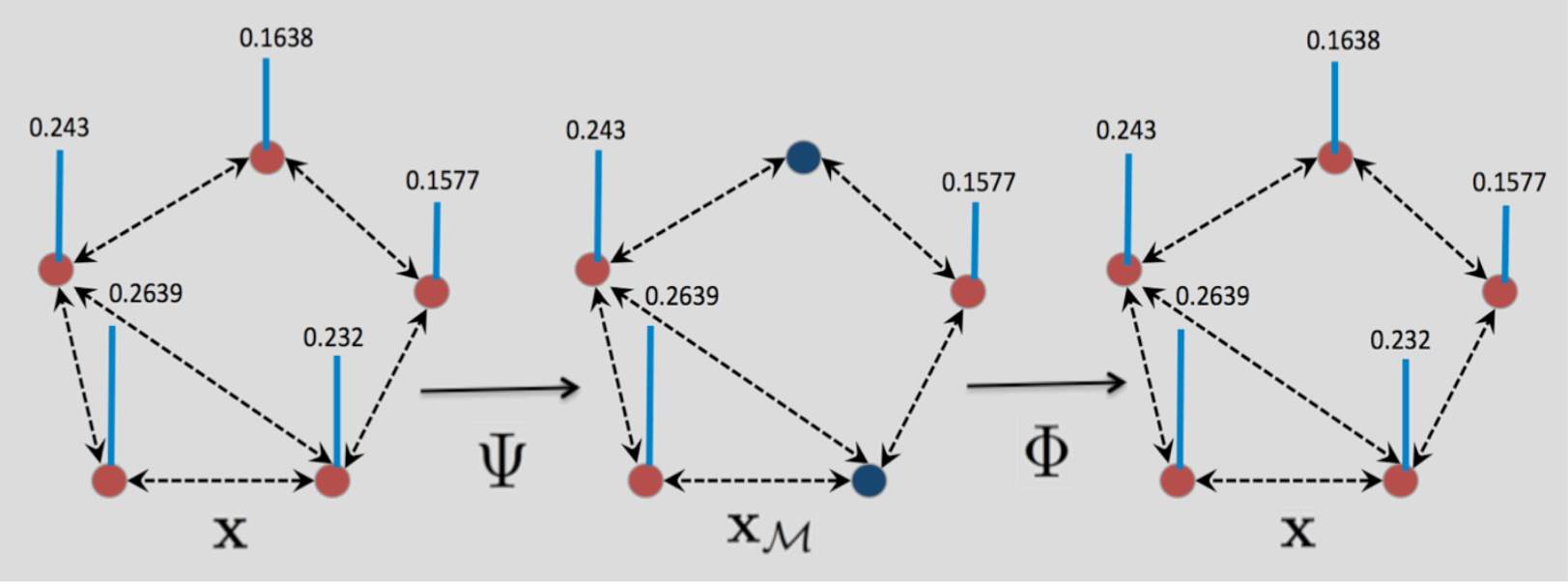


Figure 3.2: Sampling followed by interpolation. The arrows indicate different edge weights for two nodes.

We consider bandlimited graph signals here, where perfect recovery is possible.

Definition 4. A graph signal is called *bandlimited* when there exists $K \in \{0, 1, \cdots, N-1\}$ such that its graph Fourier transform $\widehat{\mathbf{x}}$ satisfies

$$\widehat{x}_i = 0 \quad \text{for all} \quad i \geq K.$$

The smallest such K is called the *bandwidth* of $\mathbf{x}$. A graph signal that is not bandlimited is called a *full-band graph signal*.

Definition 5. The set of graph signals in $\mathbb{R}^N$ with bandwidth of at most K is a closed subspace denoted $\mathrm{BL}_K(\mathbf{V}^{-1})$.

Following Theorem 5.2 in[38], we obtain the following result.

Theorem 4. Let $\mathbf{V}_{(K)}$ be the first K columns of $\mathbf{V}$ and let the sampling operator Ψ satisfy

$$\mathrm{rank}(\Psi\, \mathbf{V}_{(K)}) = K.$$

The interpolation operator $\Phi = \mathbf{V}_{(K)}\, \mathbf{U}$, with $\mathbf{U}\, \Psi\, \mathbf{V}_{(K)}$ a $K \times K$ identity matrix, where $\mathbf{U} \in \mathbb{R}^{K \times M}$, achieves perfect recovery:

$$\mathbf{x} = \Phi\Psi\mathbf{x}, \quad \text{for any } \mathbf{x} \in \mathrm{BL}_K(\mathbf{V}^{-1}).$$

Since we do not specify the ordering of frequencies, we can reorder the eigenvalues and permute the corresponding eigenvectors in the graph Fourier transform matrix to choose any band in the graph Fourier domain.

The bandlimited restriction is equivalent to requiring limited support in the graph Fourier domain.

SAMPLING OF BANDLIMITED GRAPH SIGNALS

The sample size M should be no smaller than the bandwidth K. When $M < K$, $\text{rank}(\mathbf{U}\,\Psi\,\mathbf{V}_{(K)}) \leq \text{rank}(\mathbf{U}) \leq M < K$, $\mathbf{U}\,\Psi\,\mathbf{V}_{(K)}$ can never be an identity matrix. Since $\mathbf{U}\,\Psi\,\mathbf{V}_{(K)}$ an identity matrix, $\mathbf{U}$ is the inverse of $\Psi\,\mathbf{V}_{(K)}$ when $M = K$; it is a pseudo-inverse of $\Psi\,\mathbf{V}_{(K)}$ when $M > K$, where the redundancy exists. We discuss the redundancy in Section 3.2.4. For simplicity, we only consider the case where the sample size and the bandwidth are the same, i,e., $M = K$, and $\mathbf{U}$ is invertible. When $M > K$, we simply select K from M sampled signal coefficient to ensure that the sample size and the bandwidth are the same.

From Theorem 4, we see that an arbitrary sampling operator may not lead to perfect recovery even for bandlimited graph signals. The sampling operator should select at least one set of K linearly-independent rows in $\mathbf{V}_{(K)}$. Since $\mathbf{V}$ is invertible, the column vectors in $\mathbf{V}$ are linearly independent and $\text{rank}(\mathbf{V}_{(K)}) = K$ always holds. In other words, at least one set of K linearly-independent rows in $\mathbf{V}_{(K)}$ always exists. When a sampling operator Ψ satisfies $\text{rank}(\Psi\,\mathbf{V}_{(K)}) = K$, we call it a *qualified sampling operator*. Since the graph shift $\mathbf{A}$ is given, one can find such a set independently of the graph signal. Given such a set, Theorem 4 guarantees perfect recovery of bandlimited graph signals without any approximation as in [42] and any probability constraints as in compressed sensing [44]. To find linearly-independent rows in a matrix, fast algorithms exist, such as QR decomposition; see [45,38].

SAMPLING OF BANDLIMITED GRAPH SIGNALS

In the previous part, we show that perfect recovery is possible when the graph signals are bandlimited. In the following content, we show that the sampled signal coefficients form a new graph signal, whose corresponding new graph shift is constructed from the original graph shift.

Suppose a graph signal with bandwidth K, we express it as

$$\mathbf{x} = \mathbf{V}\widehat{\mathbf{x}} = \mathbf{V}_{(K)}\widehat{\mathbf{x}}_{(K)}, \tag{3.3}$$

where $\widehat{\mathbf{x}}_{(K)} \in \mathbb{R}^K$ contains the first K signal coefficients in $\widehat{\mathbf{x}}$. Let Ψ be a sampling operator that samples M coefficients in $\mathbf{x}$ to produce $\mathbf{x}_{\mathcal{M}}$, $\Phi = \mathbf{V}_{(K)}\mathbf{U}$ be an interpolating operator, and Ψ be a sampling operator, which satisfies (3.3) in Theorem 4 to perfectly recover $\mathbf{x}$ from $\mathbf{x}_{\mathcal{M}}$. We express the graph signal as

$$\mathbf{x} = \Phi\Psi\mathbf{x} = \Phi\mathbf{x}_{\mathcal{M}} = \mathbf{V}_{(K)}\mathbf{U}\mathbf{x}_{\mathcal{M}}. \tag{3.4}$$

Since (3.3) and (3.4) hold for all $\mathbf{x} \in \mathrm{BL}_K(\mathbf{V}^{-1})$, we thus get

$$\widehat{\mathbf{x}}_{(K)} = \mathbf{U}\mathbf{x}_{\mathcal{M}}.$$

Reminding ourselves from Theorem 4 that $\mathbf{U}$ is the invertible when $M = K$, we then get

$$\mathbf{x}_{\mathcal{M}} = \mathbf{U}^{-1}\mathbf{U}\mathbf{x}_{\mathcal{M}} = \mathbf{U}^{-1}\widehat{\mathbf{x}}_{(K)}.$$

The sampled signal coefficients $\mathbf{x}_{\mathcal{M}}$ can be constructed from the frequency content $\widehat{\mathbf{x}}_{(K)}$ through $\mathbf{U}^{-1}$. In addition, the frequency content $\widehat{\mathbf{x}}_{(K)}$ can be constructed from the sampled signal coefficients $\mathbf{x}_{\mathcal{M}}$ through $\mathbf{U}$, which implies that $\mathbf{x}_{\mathcal{M}}$ is a graph signal associated with the graph Fourier transform matrix $\mathbf{U}$. Since we only use the first K frequencies, the graph shift that is associated

SAMPLING OF BANDLIMITED GRAPH SIGNALS

with $\mathbf{x}_{\mathcal{M}}$ is then

$$\mathbf{A}_{\mathcal{M}} = \mathbf{U}^{-1} \Lambda_{(K)} \mathbf{U} \in \mathbb{R}^{K \times K},$$

where $\Lambda_{(K)} \in \mathbb{R}^{K \times K}$ is a diagonal matrix that samples the first K eigenvalues of Λ. The previous discussion can be summarized as follows:

Theorem 5. Let Ψ be the sampling operator to sample K coefficients in $\mathbf{x} \in \mathrm{BL}_K(\mathbf{V}^{-1})$ to produce $\mathbf{x}_{\mathcal{M}} \in \mathbb{R}^K$ and satisfy

$$\mathrm{rank}(\Psi \mathbf{V}_{(K)}) = K.$$

Let $\mathbf{U}$ be $(\Psi \mathbf{V}_{(K)})^{-1}$. Then, $\mathbf{x}_{\mathcal{M}}$ is a graph signal associated with the graph shift

$$\mathbf{A}_{\mathcal{M}} = \mathbf{U}^{+} \Lambda_{(K)} \mathbf{U} \in \mathbb{R}^{K \times K}. \tag{3.5}$$

The graph Fourier transform of $\mathbf{x}_{\mathcal{M}}$ is

$$\widehat{\mathbf{x}}_{\mathcal{M}} = \mathbf{U} \mathbf{x}_{\mathcal{M}} \in \mathbb{R}^K.$$

The inverse graph Fourier transform is

$$\mathbf{x}_{\mathcal{M}} = \mathbf{U}^{-1} \widehat{\mathbf{x}}_{\mathcal{M}} \in \mathbb{R}^K.$$

SAMPLING OF BANDLIMITED GRAPH SIGNALS

From Theorem 5, we see that the graph shift $\mathbf{A}_{\mathcal{M}}$ is constructed by sampling the rows of the eigenvector matrix and sampling the first K eigenvalues of the original graph shift $\mathbf{A}$. We simply say that $\mathbf{A}_{\mathcal{M}}$ is "sampled" from $\mathbf{A}$, preserving certain information in the graph Fourier domain.

Since the bandwidth of $\mathbf{x}$ is K, the first K coefficients in the frequency domain are $\widehat{\mathbf{x}}_{(K)} = \widehat{\mathbf{x}}_{\mathcal{M}}$, and the other $N - K$ coefficients are $\widehat{\mathbf{x}}_{(-K)} = 0$; in other words, the frequency contents are equivalent for the original graph signal $\mathbf{x}$ and the sampled graph signal $\mathbf{x}_{\mathcal{M}}$ after performing their corresponding graph Fourier transforms.

Similarly to Theorem 4, by reordering the eigenvalues and permuting the corresponding eigenvectors in the graph Fourier transform matrix, Theorem 5 is applicable for all graph signals that have limited supports in the graph Fourier domain, and the sampled graph shift $\mathbf{A}_{\mathcal{M}}$ supports the sampled signal coefficients, preserving the corresponding frequency content.

3.1.2 Finite Discrete-time Cases

We call the graph that supports a finite discrete-time signal as the *finite discrete-time graph*, which is represented by the cyclic permutation matrix[38,21],

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} 0 & 0 & \cdots & 1 \\ 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 1 & 0 \end{bmatrix} \\ &= \mathbf{V}\Lambda\mathbf{V}^{-1}, \end{aligned} \tag{3.6}$$

SAMPLING OF BANDLIMITED GRAPH SIGNALS

where the eigenvector matrix

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_0 & \mathbf{v}_1 & \cdots & \mathbf{v}_{N-1} \end{bmatrix} = \left[\tfrac{1}{\sqrt{N}} (w^{jk})^* \right]_{j,k=0,\cdots N-1}, \tag{3.7}$$

is the Hermitian transpose of the N-point discrete Fourier transform matrix, i.e., $\mathbf{V} = \mathrm{F}^*$, where $*$ is the Hermitian transpose, and $\mathbf{V}^{-1}$ is the N-point discrete Fourier transform matrix (F), i.e., $\mathbf{V}^{-1} = \mathrm{F}$, and the eigenvalue matrix is

$$\Lambda = \mathrm{diag} \begin{bmatrix} \lambda_0 & \lambda_1 & \cdots & \lambda_{N-1} \end{bmatrix}, \tag{3.8}$$

where $\lambda_i = w^i$, $w = e^{-2\pi j/N}$. We see that Definitions 6, 7 and Theorem 4 are immediately applicable to finite discrete-time signals.

Definition 6. A discrete-time signal is called *bandlimited* when there exists $K \in \{0, 1, \cdots, N-1\}$ such that its discrete Fourier transform $\widehat{\mathbf{x}}$ satisfies

$$\widehat{x}_i = 0 \quad \text{for all} \quad i \geq K.$$

The smallest such K is called the *bandwidth* of $\mathbf{x}$. A discrete-time signal that is not bandlimited is called a *full-band discrete-time signal*.

SAMPLING OF BANDLIMITED GRAPH SIGNALS

Definition 7. The set of discrete-time signals in $\mathbb{R}^N$ with bandwidth of at most K is a closed subspace denoted $\mathrm{BL}_K(\mathrm{F})$, with F as the discrete Fourier transform matrix.

With this definition of the discrete Fourier transform matrix, the highest frequency is in the middle of the spectrum (although this is just a matter of ordering). From Definitions 6 and 7, we can permute the rows in the discrete Fourier transform matrix to choose any frequency band. Since the discrete Fourier transform matrix is a Vandermonde matrix, any K rows of $\mathrm{F}^*_{(K)}$ are independent[45,38]; in other words, $\mathrm{rank}(\Psi\,\mathrm{F}^*_{(K)}) = K$ always hold when $M \geq K$. We apply now Theorem 4 to obtain the following result.

Theorem 6. Let $\mathrm{F}^*_{(K)}$ be the first K columns of F^* and let the sampling operator Ψ satisfy the sampling number M is no less than the bandwidth K. The interpolation operator $\Phi = \mathrm{F}^*_{(K)}\,\mathbf{U}$, with $\mathbf{U}\,\Psi\,\mathrm{F}^*_{(K)}$ a $K \times K$ identity matrix, achieves perfect recovery,

$$\mathbf{x} = \Phi\Psi\mathbf{x}, \quad \text{for any } \mathbf{x} \in \mathrm{BL}_K(\mathrm{F}).$$

From Theorem 6, we can perfectly recover a discrete-time signal when it is bandlimited.

Similarly to Theorem 5, we can show that a new graph shift can be constructed from the finite discrete-time graph. Multiple sampling mechanisms can be done to sample a new graph shift, to obtain an intuitive one, we do as follows. Suppose $\mathbf{x} \in \mathbb{R}^N$ is a finite discrete-time signal, where N is even, and the corresponding finite discrete-time graph is represented by the cyclic permutation matrix, $\mathbf{A}$, as in (3.6). We reorder the frequencies in (3.8), by putting the frequencies with even indices first as

$$\widetilde{\Lambda} = \mathrm{diag}\begin{bmatrix}\lambda_0 & \lambda_2 & \cdots & \lambda_{N-2} & \lambda_1 & \lambda_3 & \cdots & \lambda_{N-1}\end{bmatrix},$$

Correspondingly, we reorder the columns of $\mathbf{V}$ in (3.7) by putting the columns with even indices

SAMPLING OF BANDLIMITED GRAPH SIGNALS

first as

$$\widetilde{\mathbf{V}} = \begin{bmatrix} \mathbf{v}_0 & \mathbf{v}_2 & \cdots & \mathbf{v}_{N-2} & \mathbf{v}_1 & \mathbf{v}_3 & \cdots & \mathbf{v}_{N-1} \end{bmatrix}.$$

One can check that $\widetilde{\mathbf{V}}\widetilde{\Lambda}\widetilde{\mathbf{V}}^{-1}$ is still the same cyclic permutation matrix, where $\widetilde{\mathbf{V}}^{-1}$ is the inverse of $\widetilde{\mathbf{V}}$. Suppose we want to preserve the first $N/2$ frequency contents in $\widetilde{\Lambda}$, the sampled frequencies are then

$$\widetilde{\Lambda}_{(N/2)} = \operatorname{diag}\begin{bmatrix} \lambda_0 & \lambda_2 & \cdots & \lambda_{N-2} \end{bmatrix}.$$

Let a sampling operator Ψ choose the first $N/2$ rows in $\widetilde{\mathbf{V}}_{(N/2)}$,

$$\Psi\widetilde{\mathbf{V}}_{(N/2)} = \left[\tfrac{1}{\sqrt{N}}(w^{2jk})^*\right]_{j,k=0,\cdots N/2-1},$$

which is the Hermitian transpose of the $N/2$ discrete Fourier transform and satisfies $\operatorname{rank}(\Psi\widetilde{\mathbf{V}}_{(N/2)}) = N/2$ in Theorem 5. The sampled graph Fourier transform matrix $\mathbf{U} = (\Psi\widetilde{\mathbf{V}}_{(N/2)}))^{-1}$ is the $N/2$ discrete Fourier transform. The sampled graph shift is then constructed as

$$\mathbf{A}_{\mathcal{M}} = \mathbf{U}^{-1}\,\widetilde{\Lambda}_{(N/2)}\,\mathbf{U},$$

which is exactly the $N/2 \times N/2$ cyclic permutation matrix. Hence, we have shown that by choosing an appropriate sampling mechanism, a smaller finite discrete-time graph is obtained from a larger finite discrete-time graph by using Theorem 5. We note that using a different ordering or sampling operator, would result in a graph shift that can be different and non-intuitive. This is however a matter of choosing different frequency contents.

SAMPLING OF BANDLIMITED GRAPH SIGNALS

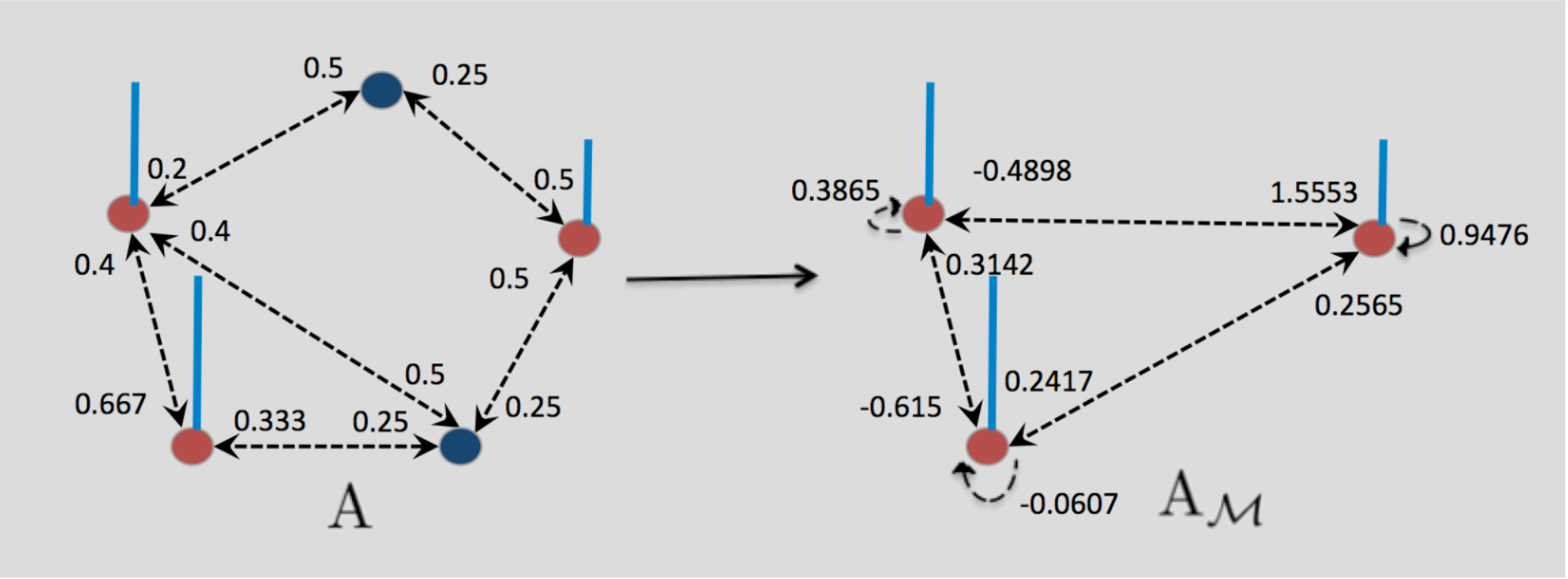


Figure 3.3: Sampling a graph.

SAMPLING OF BANDLIMITED GRAPH SIGNALS

We consider a five-node directed graph with graph shift

$$\mathbf{A} = \begin{bmatrix} 0 & 0.4 & 0.4 & 0 & 0.2 \\ 0.667 & 0 & 0.333 & 0 & 0 \\ 0.5 & 0.25 & 0 & 0.25 & 0 \\ 0 & 0 & 0.5 & 0 & 0.5 \\ 0.5 & 0 & 0 & 0.5 & 0 \end{bmatrix}.$$

The corresponding inverse graph Fourier transform matrix is

$$\mathbf{V} = \begin{bmatrix} 0.4472 & 0.1936 & 0.253 & 0.3532 & -0.4026 \\ 0.4472 & 0.4034 & 0.1604 & -0.7446 & 0.1782 \\ 0.4472 & 0.0842 & -0.5618 & 0.2862 & 0.362 \\ 0.4472 & -0.6596 & -0.4053 & -0.4706 & -0.5733 \\ 0.4472 & -0.598 & 0.656 & 0.132 & 0.5886 \end{bmatrix},$$

and the frequencies are

$$\Lambda = \text{diag}\begin{bmatrix} 1 & 0.3895 & -0.1161 & -0.444 & -0.829 \end{bmatrix}.$$

We generate a bandlimited graph signal $\mathbf{x} \in \text{BL}_3(\mathbf{V}^{-1})$ as

SAMPLING OF BANDLIMITED GRAPH SIGNALS

We generate a bandlimited graph signal $\mathbf{x} \in \mathrm{BL}_3(\mathbf{V}^{-1})$ as

$$\mathbf{x} = \begin{bmatrix} 0.242 & 0.2639 & 0.232 & 0.1577 & 0.1638 \end{bmatrix}^T.$$

We can check the first three columns of $\mathbf{V}$ to see that all sets of three rows are independent. According to the sampling theorem, we can recover $\mathbf{x}$ perfectly by sampling any three of its coefficients; for example, sample the first,second and the fourth coefficients. Then, $\mathcal{M} = \{1, 2, 4\}$, $\mathbf{x}_{\mathcal{M}} = \begin{bmatrix} 0.242 & 0.2639 & 0.1577 \end{bmatrix}^T$, and the sampling operator

$$\Psi = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

We recover $\mathbf{x}$ by using the following interpolation operator (see Figure 3.2)

$$\Phi = \mathbf{V}_{(3)}(\Psi\,\mathbf{V}_{(3)})^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -2.7043 & 2.8703 & 0.834 \\ 0 & 0 & 1 \\ 5.0363 & -3.9845 & -0.0518 \end{bmatrix}.$$

SAMPLING OF BANDLIMITED GRAPH SIGNALS

The inverse graph Fourier transform matrix is

$$\mathbf{U}^{-1} = \Psi\,\mathbf{V}_{(3)} = \begin{bmatrix} 0.4472 & 0.1936 & 0.253 \\ 0.4472 & 0.4034 & 0.1604 \\ 0.4472 & -0.6596 & -0.4053 \end{bmatrix},$$

and the sampled frequencies are

$$\Lambda_{(3)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.3895 & 0 \\ 0 & 0 & -0.1161 \end{bmatrix},$$

The sampled graph shift is then constructed as

$$\mathbf{A}_{\mathcal{M}} = \mathbf{U}^{-1}\,\Lambda_{(3)}\,\mathbf{U} = \begin{bmatrix} 0.3865 & 0.3142 & 0.2417 \\ -0.615 & -0.0607 & -0.4898 \\ 1.5553 & 0.2565 & 0.9476 \end{bmatrix}.$$

We see that while the sampled graph shift contains self-loops and negative weights, which seems to be dissimilar to $\mathbf{A}$, $\mathbf{A}_{\mathcal{M}}$ perfectly preserves the frequency content of $\mathbf{A}$.

SAMPLING OF BANDLIMITED GRAPH SIGNALS

In view of signal processing on graphs, the eigenvalues $\{\lambda_k\}$ of the Laplacian are regarded as frequencies and the corresponding eigenvectors $\{\mathbf{u}_k\}$ are regarded as basis vectors. Consider an arbitrary graph signal $\mathbf{f} \in \Re^N$. Its frequency component corresponding to λ_k is the inner product between $\mathbf{f}$ and the eigenvector $\mathbf{u}_k$, denoted as

$$\hat{f}(\lambda_k) = \langle \mathbf{f}, \mathbf{u}_k \rangle = \sum_{i=1}^{N} f(i) u_k(i).$$

SAMPLING OF BANDLIMITED GRAPH SIGNALS

The eigenvectors associated with small eigenvalues have similar values on neighboring vertices, while the eigenvectors associated with large eigenvalues are the opposite. As a result, the frequency components associated with small and large eigenvalues correspond to the low-frequency and high-frequency parts of the signal, respectively [1, 33].

Suppose $\mathbf{f} \in \Re^N$ is a graph signal on an N-vertex graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$. We say $\mathbf{f}$ is ω-bandlimited if its frequency components corresponding to eigenvalues larger than ω are all zero. In other words, the spectral support of $\mathbf{f}$ is a subset of $[0, \omega]$. The subspace consisting of all ω-bandlimited signals on graph $\mathcal{G}$ is called the Paley-Wiener space, which is a Hilbert space and denoted as $PW_\omega(\mathcal{G})$.

Suppose that for $\mathbf{f} \in PW_\omega(\mathcal{G})$ only the entries on a selected set of nodes $\{f(u)\}_{u \in \mathcal{S}}$ are known, where $\mathcal{S} \subseteq \mathcal{V}$ is the sampled vertex set. The sampling and reconstruction problem is to recover the ω-bandlimited original signal $\mathbf{f}$ based on the sampled data $\{f(u)\}_{u \in \mathcal{S}}$.

DISTRIBUTED SIGNAL RECONSTRUCTION

3 Distributed Reconstruction of Time-varying Bandlimited Graph Signal

3.1 Motivation

We consider the distributed reconstruction of a time-varying low-frequency signal defined over graph by sampling at a small portion of nodes. The problem could be described in the scenario of wireless sensor network (WSN). For a given WSN, there are unknown function $f_*^{(t)}(v)$ associating with node v at time t. Suppose that the function is slowly varying over both the time domain and the space domain. A snapshot of such function could be modeled as an unknown time-varying low-frequency signal $\mathbf{f}_*^{(t)} \in PW_\omega(\mathcal{G})$ located over a graph.[1]

Suppose that the WSN is a hybrid network and only a small subset of nodes in $\mathcal{S}$ are equipped with sensors. As a result, one can measure the signal entries on support $\mathcal{S}$ as $\{f_*^{(t)}(u)\}_{u\in\mathcal{S}}$ at time t. Our purpose is to distributivedly estimate the function values at all nodes $\{f_*^{(t)}(v)\}_{v\in\mathcal{V}}$, by using historical measurements at selected nodes $\{f_*^{(\tau)}(u)\}_{u\in\mathcal{S},\tau\leq t}$. See Fig. 1 for a demonstration of the raised problem.

DISTRIBUTED SIGNAL RECONSTRUCTION

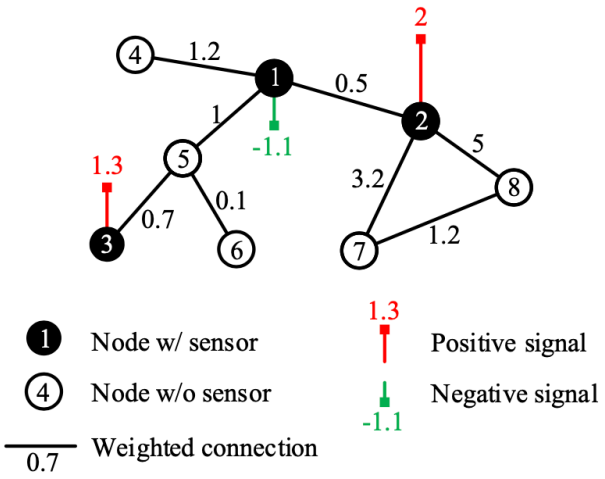


Figure 1: An example of graph signal reconstruction over wireless sensor network.

When the signal is time-invariant, the problem reduces to bandlimited graph signal reconstruction, where the nodes with and without sensors correspond to sampled and missing data. In the distributed setting, the centralized iteration (2) no longer applies, as it is impossible for every node to obtain the instant estimation errors $\{f_*(u) - f^{(k)}(u)\}_{u\in\mathcal{S}}$.

In what follows, we focus on the generalization of ILSR method to distributed systems and time-varying signals. We proposed an algorithm called distributed least square reconstruction (DLSR). By letting each node conducting the iteration locally at each time instant, DLSR can adaptively reconstruct the missing entries of a slowly time-varying graph signal.

3.2 Algorithm Description

The basic idea of DLSR is to spread the current estimation errors associated with the representative nodes (which are equipped with sensors) to all other nodes over the connected network. Every node iteratively updates its own estimation based on its received messages.

The driver of the proposed algorithm is on those nodes with sensors, which calculates the error between the measurement $f_*^{(k)}(u)$ and the temporary estimation $f^{(k)}(u)$ in the kth iteration by

$$\epsilon^{(k)}(u) = f_*^{(k)}(u) - f^{(k)}(u), \quad \forall u \in \mathcal{S}.$$

Then the estimation errors at node u ($u \in \mathcal{S}$) are transmitted to other nodes in the network. At the kth iteration, an arbitrary v collects a set of delayed but most recent estimation errors,

$$\{\epsilon^{(k-\tau(u,v))}(u)\}_{u\in\mathcal{S}}, \quad \forall v \in \mathcal{V}(\mathcal{G}),$$

where $\tau(u, v)$ denotes the transmission delay from node u to node v [2]. We denote the maximal transmission delay of the network by

$$\tau = \max_{u\in\mathcal{S}, v\in\mathcal{V}(\mathcal{G})} \tau(u, v).$$

DISTRIBUTED SIGNAL RECONSTRUCTION

Table 1: DLSR Algorithm at Representative Node $u \in \mathcal{S}$.

Parameter: $\mathcal{S}, \{\tau(u', u)\}_{u', u \in \mathcal{S}}, \mu_k, \beta_k$;

Initialization: $f^{(0)}(u) = 0$, calculate $(\mathcal{P}_\omega \boldsymbol{\delta}_{u'})(u), \forall u' \in \mathcal{S}$;

For $k = 0, 1, 2, \cdots$

1) Input: $f_*^{(k)}(u)$;

2) Estimation:

$\epsilon^{(k)}(u) = f_*^{(k)}(u) - f^{(k)}(u)$;

3) Communication:

Send $\epsilon^{(k)}(u)$ and $\epsilon^{(k-1-\tau(u',u))}(u')$ to neighbors, $\forall u' \in \mathcal{S} \backslash u$;

Receive $\epsilon^{(k-\tau(u',u))}(u')$ from neighbors, $\forall u' \in \mathcal{S} \backslash u$;

4) Update Storage:

Save $\epsilon^{(k-\tau(u',u))}(u')$, $\forall u' \in \mathcal{S} \backslash u$;

5) Update Estimation:

$$f^{(k+1)}(u) = (1 - \mu_{k+1}\beta_{k+1}) f^{(k)}(u) + \mu_{k+1} \sum_{u' \in \mathcal{S}} \epsilon^{(k-\tau(u',u))}(u') (\mathcal{P}_\omega \boldsymbol{\delta}_{u'})(u).$$

End

Utilizing the most recent estimation errors, node v updates its local estimate by

$$f^{(k+1)}(v) = (1 - \mu_{k+1}\beta_{k+1}) f^{(k)}(v) + \mu_{k+1} \sum_{u \in \mathcal{S}} \epsilon^{(k-\tau(u,v))}(u) (\mathcal{P}_\omega \boldsymbol{\delta}_u)(v), \quad \forall v \in \mathcal{V}(\mathcal{G}) \quad (3)$$

where μ_{k+1} and β_{k+1} denote the stepsize and decay factor, respectively. $(\mathcal{P}_\omega \boldsymbol{\delta}_u)(v)$ denotes the entry at v of the lowpass component of $\boldsymbol{\delta}_u$, which could be calculated and stored before the system starts.

Table 1: The frames in space $PW_\omega(\mathcal{G}), \forall \omega < 1/Q_{\max}^2$, and their bounds.

Frame	Lower bound	Upper bound
$\{\mathcal{P}_\omega(\boldsymbol{\delta}_u)\}_{u \in \mathcal{S}}$	$(1-\gamma)^2/N_{\max}$	1
$\{\sqrt{\lvert\mathcal{N}(u)\rvert}\mathcal{P}_\omega(\boldsymbol{\delta}_u)\}_{u \in \mathcal{S}}$	$(1-\gamma)^2$	$(1+\gamma)^2$
$\{\mathcal{P}_\omega(\boldsymbol{\delta}_{\mathcal{N}(u)})\}_{u \in \mathcal{S}}$	$(1-\gamma)^2$	$N_{\max}$

DISTRIBUTED SIGNAL RECONSTRUCTION

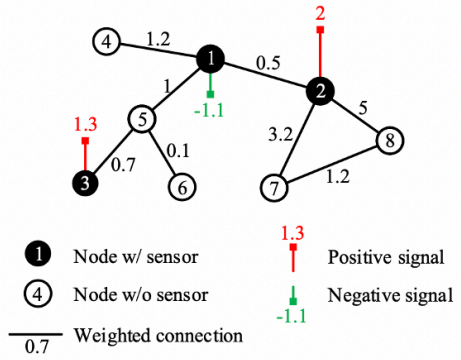


Figure 1: An example of graph signal reconstruction over wireless sensor network.

3.3 An Example

In order to provide some intuition for our distributed algorithm, let us refer to Fig. 1 and describe what happens on a typical node in the network.

- As a representative node equipped with sensor, node 1 will get a measurement $f_*^{(k)}(1)$ at the kth iteration and then calculate the estimation error $\epsilon^{(k)}(1)$. The estimation error will be send to its neighbors of node $2, 4$, and 5, and then forwarded to others. At the same slot, node 1 will receive the estimation errors of other nodes with sensors and use the most recent ones ($\epsilon^{(k-1)}(2)$ from node 2 and

DISTRIBUTED SIGNAL RECONSTRUCTION

Table 2: DLSR Algorithm at Non-representative Node $v \in \mathcal{V}(\mathcal{G})\backslash\mathcal{S}$.

Parameter: $\mathcal{S}, \{\tau(u,v)\}_{u\in\mathcal{S},v\in\mathcal{V}(\mathcal{G})\backslash\mathcal{S}}, \mu_k, \beta_k$;

Initialization: $f^{(0)}(v) = 0$, calculate $(\mathcal{P}_\omega \boldsymbol{\delta}_u)(v), \forall u \in \mathcal{S}$;

For $k = 0, 1, 2, \cdots$

1) Communication:

Send $\epsilon^{(k-1-\tau(u,v))}(u)$ to neighbors, $\forall u \in \mathcal{S}$;

Receive $\epsilon^{(k-\tau(u,v))}(u)$ from neighbors, $\forall u \in \mathcal{S}$;

2) Update Storage:

Save $\epsilon^{(k-\tau(u,v))}(u)$, $\forall u \in \mathcal{S}$;

3) Update Estimation:

$$f^{(k+1)}(v) = (1 - \mu_{k+1}\beta_{k+1}) f^{(k)}(v) + \mu_{k+1} \sum_{u\in\mathcal{S}} \epsilon^{(k-\tau(u,v))}(u)(\mathcal{P}_\omega \boldsymbol{\delta}_u)(v);$$

4) Output: $f^{(k+1)}(v)$.

End

DISTRIBUTED SIGNAL RECONSTRUCTION

$\epsilon^{(k-2)}(3)$ from node 5). Consequently, it could update the estimation by

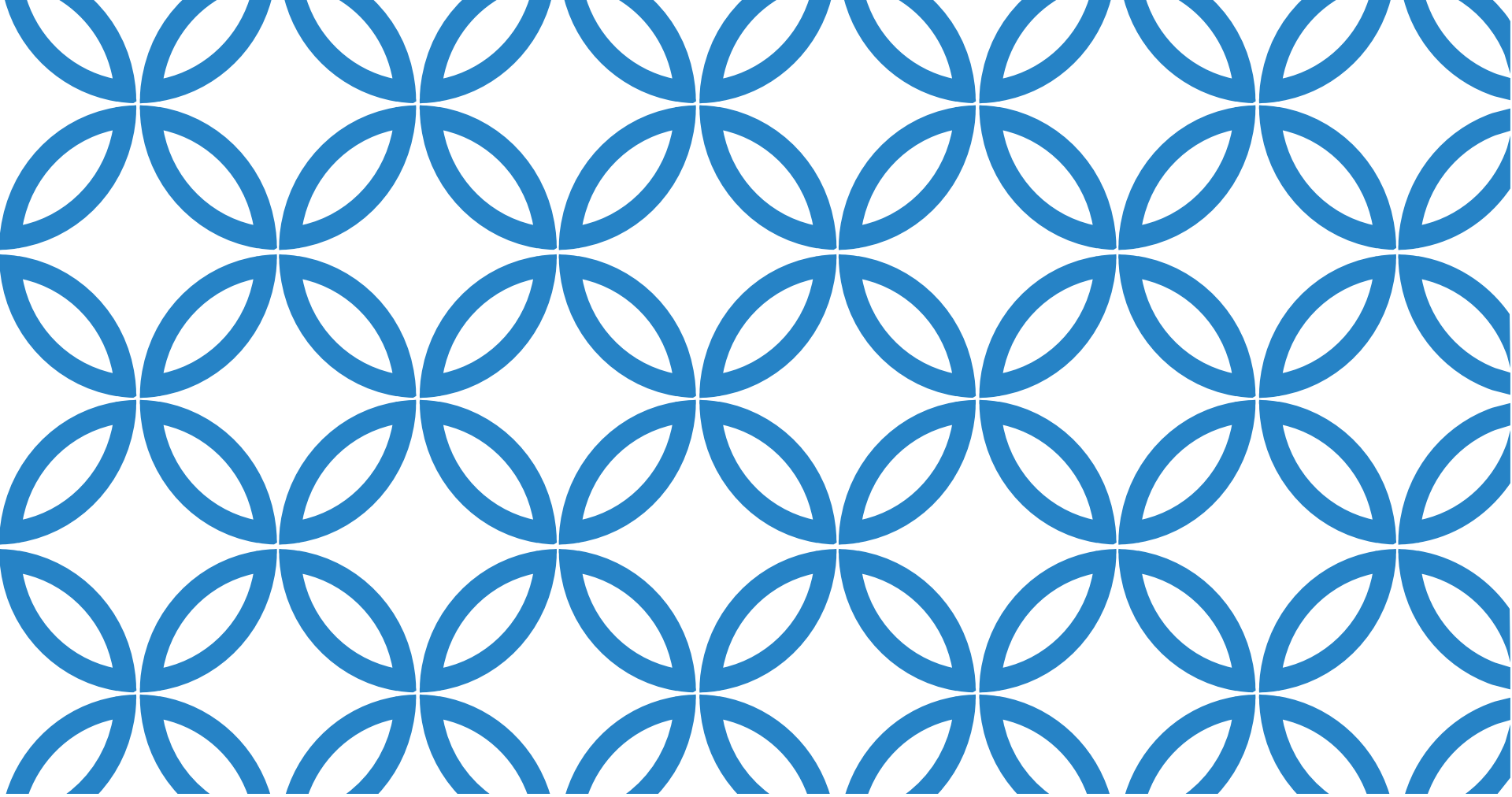
$$f^{(k+1)}(1) = (1-\mu_{k+1}\beta_{k+1})f^{(k)}(1) + \mu_{k+1}\Big(\epsilon^{(k)}(1)\cdot(\mathcal{P}_\omega\boldsymbol{\delta}_1)(1) + \epsilon^{(k-1)}(2)(\mathcal{P}_\omega\boldsymbol{\delta}_2)(1) + \epsilon^{(k-2)}(3)(\mathcal{P}_\omega\boldsymbol{\delta}_3)(1)\Big).$$

One may notice that the new estimation is not the output of the proposed algorithm, because our purpose is to estimate the strength of the signal associated with the node without sensor. However, the new estimation errors at representative nodes will be transmitted over the network to help all others to conduct their estimation.

- As a regular node that is not equipped with sensor, at the kth iteration, node 5 will receive $\epsilon^{(k-1)}(1), \epsilon^{(k-2)}(2)$, and $\epsilon^{(k-1)}(3)$ from its neighbors. Then it will transmit the most recent estimation errors of nodes with sensors to its neighbors $1, 3$, and 6. The estimate of node 5 is updated by

$$f^{(k+1)}(5) = (1-\mu_{k+1}\beta_{k+1})f^{(k)}(5) + \mu_{k+1}\Big(\epsilon^{(k-1)}(1)(\mathcal{P}_\omega\boldsymbol{\delta}_1)(5) + \epsilon^{(k-2)}(2)(\mathcal{P}_\omega\boldsymbol{\delta}_2)(5) + \epsilon^{(k-3)}(3)(\mathcal{P}_\omega\boldsymbol{\delta}_3)(5)\Big).$$

The new estimate will be sent out as a temporary result of the proposed algorithm.



DISTRIBUTED KALMAN FILTER

REFERENCES

Distributed Kalman Filter with Embedded Consensus Filters

R. Olfati-Saber

DISTRIBUTED KALMAN FILTERS IN SENSOR NETWORKS: BIPARTITE FUSION GRAPHS

Usman A. Khan and José M. F. Moura

Carnegie Mellon University
Department of Electrical and Computer Engineering
Pittsburgh, PA 15213
{ukhan, moura}@ece.cmu.edu

CENTRALIZED KF

With sensor networks the observations of the field of interest are distributed across different sensors.

All these observations are to be incorporated in the implementation of the Kalman filter to ensure optimal performance. Collecting these observations at a single location (fusion center) implements a centralized Kalman filter.

The fusion center then communicates the estimates back to the sensors. In large-scale dynamical systems, the centralized Kalman filter is impractical because it requires:

(i) long-distance communication since the sensors span a large geographical area; and

(ii) high computation because the state-space models coming from such large-scale systems have high-dimensional state vectors. Furthermore, a centralized scheme has the disadvantages of large latency and a single point of failure.

CONSENSUS ESTIMATION

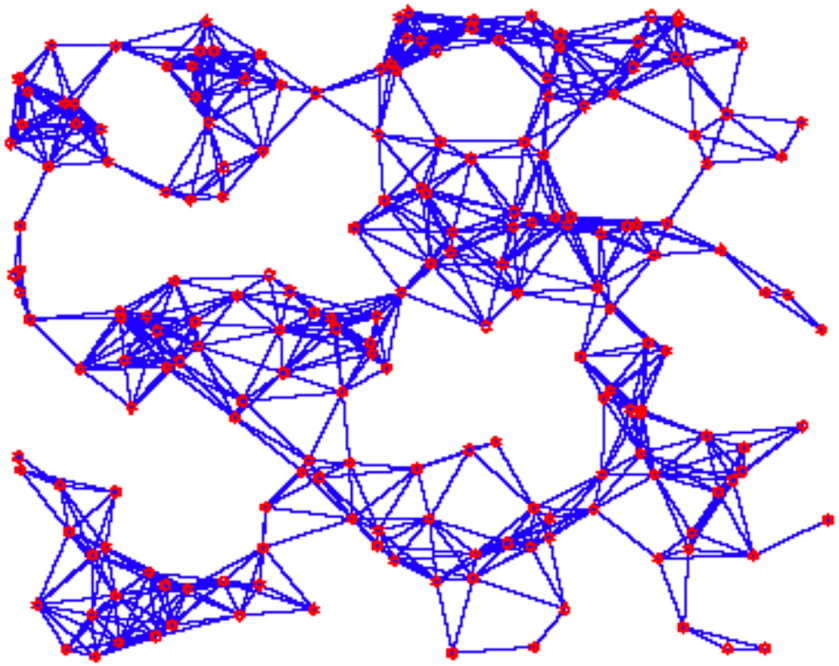
- The problem of distributed Kalman filtering (DKF) for sensor networks is one of the most fundamental distributed estimation problems for scalable sensor fusion.
- DKF problem as two separate dynamic consensus problems
  - weighted measurements and
  - inverse-covariance matrices.
- These to data fusion problems are solved in a distributed way using lowpass and band-pass consensus filters.

CONSENSUS FILTERS

- Consensus filters are distributed algorithms that allow calculation of average-consensus of time-varying signals.
- The central Kalman filter for sensor networks can be decomposed into n micro-Kalman filters with inputs that are provided by two types of consensus filters.
- This network of micro-Kalman filters collectively are capable to provide an estimate of the state of the process (under observation) that is identical to the estimate obtained by a central Kalman filter given that all nodes agree on two central sums.

PROBLEM FORMULATION

- Consider a sensor network with n sensors that are interconnected via an overlay network G (e.g. a connected undirected graph).



DISTRIBUTED KALMAN FILTER

- Distributed estimation and filtering is one of the most fundamental collaborative information processing problems in wireless sensor networks (WSN).
- We have a number of sensors observing a process which is not observable for each sensor, but the process is observable for the collection of sensors.
- The Central Kalman Filter estimate (x_c) is computationally expensive.
- *Is it possible that each sensor estimate of x_c based on only local information from its neighbors?*

FILTER CONSTRUCTION

- Let $z(k) = col(z_1(k), \cdots, z_n(k)) \in \mathbb{R}^{np}$ be the collective sensor data of the entire sensor network at time k.
- Given the information $Z_k = \{z(0), \cdots, z(k)\}$, we want to estimate the state of the process.
- Define
  - estimate of the process state: $\hat{x}_k = E(x_k|Z_k), \bar{x}_k = E(x_k|Z_{k-1})$
  - estimate of the error covariance: $P_k = \Sigma_{k|k-1}$, $M_k = \Sigma_{k|k}$
- Thus we want to perform KF for the system:
  - $x(k+1) = A_k x(k) + B_k w(k)$
  - $z(k) = H_k x(k) + v_k$
  - with $H_k = col(H_1(k), \ldots, H_n(k))$, $v_k = col(v_1(k), \ldots, v_n(k))$, $R_k = diag(R_1(k), \ldots, R_n(k))$

FILTER CONSTRUCTION

Kalman Filter iterations for the sensor network would be of the form:

- $M_k = (P_k^{-1} + H_k^\top R_k^{-1} H_k)^{-1}$
- $K_k = M_k H_k^\top R_k^{-1}$
- $\hat{x}(k) = \bar{x}(k) + K_k(z(k) - H_k \bar{x}(k))$: central estimate $(\hat{x}_c)$
- $P(k+1) = A_k M_k A_k^\top + B_k Q_k B_k^\top$
- $\bar{x}(k+1) = A_k \hat{x}(k)$

Next: Perform distributed state estimation (or tracking) for the process

FILTER DISTRIBUTION

Define two aggregate quantities:

- Fused inverse-covariance matrices:
  $S(k) = 1/n \sum_i H_i^\top(k) R_i^{-1}(k) H_i(k)$
- Fused sensor data: $y(k) = 1/n \sum_i H_i^\top(k) R_i^{-1}(k) z_i(k)$

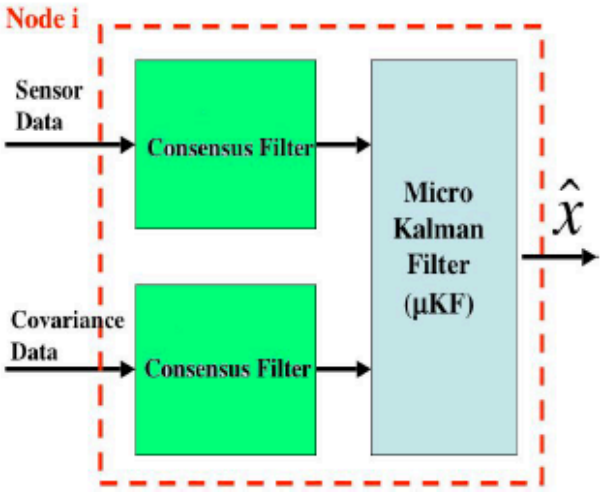
Transformed update equations for the Central KF:

- $M_\mu(k) = (P_\mu^{-1}(k) + S(k))^{-1}$
- $\hat{x}(k) = \bar{x}(k) + M_\mu(k)(y(k) - S(k)\bar{x}(k))$
- $P_\mu(k+1) = A_k M_\mu(k) A_k^\top + B_k Q_\mu(k) B_k^\top$
- $\bar{x}(k+1) = A_k \hat{x}(k)$
- where $M_\mu(k) = nM_k$, $Q_i(k) = nQ(k)$, $P_\mu(0) = nP_0$.

If each sensor implements a KF with above iterations, then all nodes have the same estimates as central estimate. Is it a DKF?

FILTER DISTRIBUTION

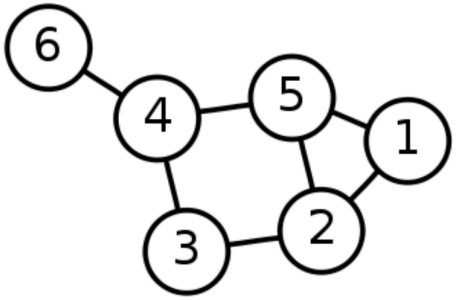
- If each node can compute the averages $y(k)$ and $S(k)$, a distributed KF emerges!
- Two consensus filters to compute $S(k)$ and $y(k)$ at each node using local information.
- Each node of the distributed Kalman filter that provides a state estimate is called a Micro-Filter.
- Microfilters have identical structures.



Laplacian matrix

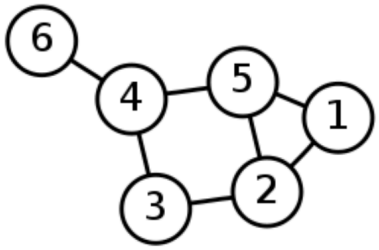
6n-graf.svg (SVG file, nominally 333 × 220 pixels, file size: 3 KB)

$$\ell_{i,j} := \begin{cases} \deg(v_i) & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise.} \end{cases}$$



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

- The **algebraic connectivity** of a graph G is the second-smallest eigenvalue of the Laplacian matrix of G.[1] This eigenvalue is greater than 0 if and only if G is a connected graph.
- This is a corollary to the fact that the number of times 0 appears as an eigenvalue in the Laplacian is the number of connected components in the graph.
- The magnitude of this value reflects how well connected the overall graph is, and has been used in analysing the synchronizability of networks.



An example graph, with 6 vertices, diameter 3, connectivity 1, and algebraic connectivity 0.722

The **diameter** of a connected graph is the distance between the two vertices which are furthest from each other.

Let $N_i = \{j : (i,j) \in E\}$ be the set of neighbors of node i on graph G. Moreover, let $L = D - \mathcal{A}$ be the Laplacian matrix of G and $\lambda_2 = \lambda_2(L)$ denote its algebraic connectivity. The high-pass consensus filter is a linear system in the form

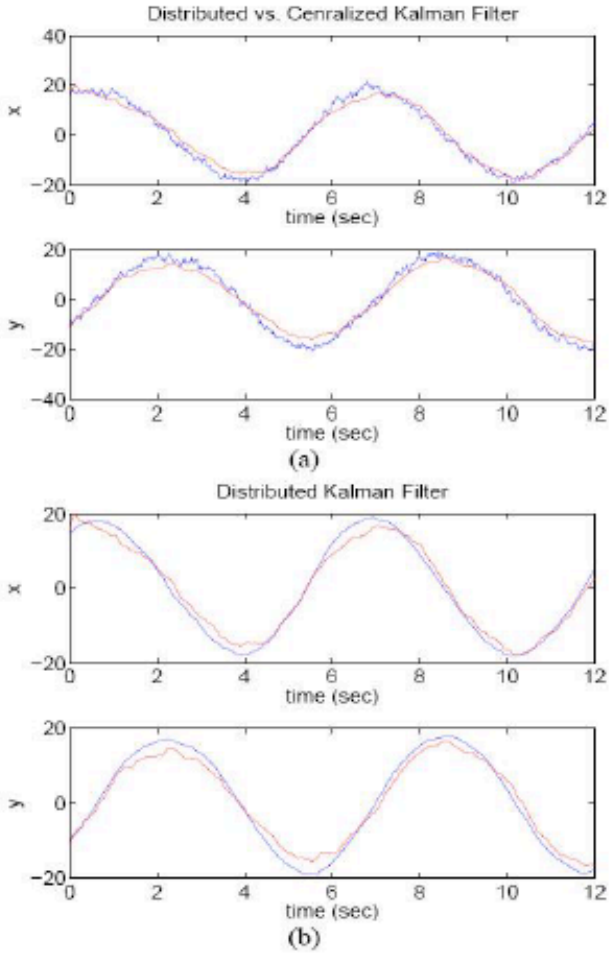
$$\begin{cases} \dot{q}_i = \beta \sum_{j \in N_i} (q_j - q_i) + \beta \sum_{j \in N_i} (u_j - u_i); \ \beta > 0 \\ y_i = q_i + u_i \end{cases} \tag{11}$$

where u_i is the input of node i, q_i is the state of the consensus filter, and y_i is its output. The gain $\beta > 0$ is relatively large ($\beta \sim O(1/\lambda_2)$) for randomly generated ad hoc topologies

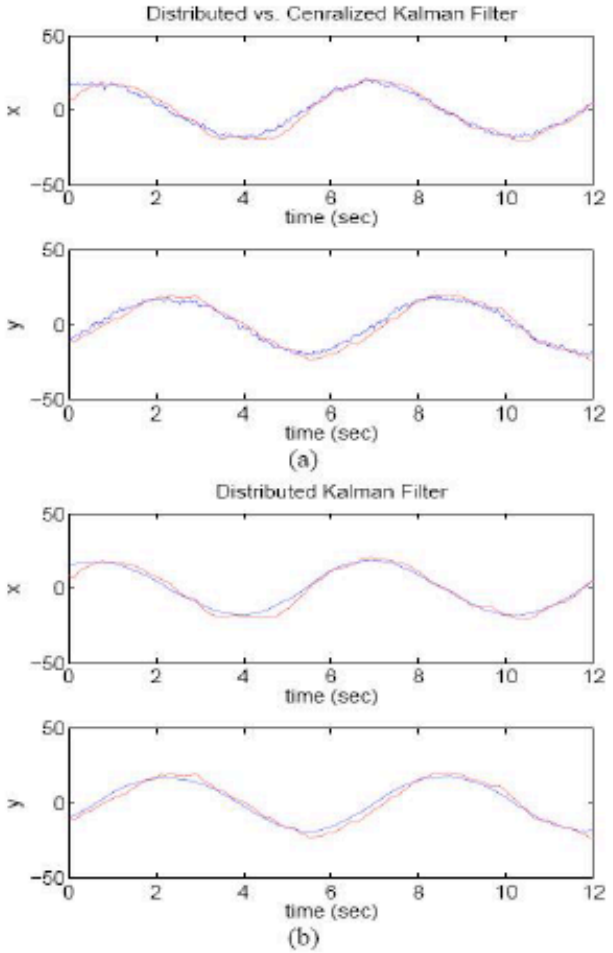
DISTRIBUTED KALMAN FILTER

- Use highpass consensus filters of the form
  - $\dot{q}_i = \beta \sum_{j \in N_i}(q_j - q_i) + \beta \sum_{j \in N_i}(u_j - u_i),\ \ \beta > 0$
  - $p_i = q_i + u_i$
  - where $\beta \sim O(1/\lambda_2)$ is relatively large.
- u denotes the input for each node:
  - For CF1 [$\rightarrow y(k)$]: $u_j = H_j^\top R_j^{-1} z_j$
  - For CF2 [$\rightarrow S(k)$]: $u_j = H_j^\top R_j^{-1} H_j$
- It is shown that for a connected network, outputs $p_i^1(k)$ and $p_i^2(k)$ of the highpass consensus filters asymptotically converge to $y(k)$ and $S(k)$.

- Node i sends the message:
  $msg_i = (q_i^1(k), q_i^2(k), u_i^1(k), u_i^2(k))$ to all of its neighbors. $\Rightarrow$ Message size is of dimension $O(m(m+1))$ with m being the dimension of the state x.
- The communication scheme is fully compatible with packet-based communication in real-world WSN.



Distributed position estimation for a moving object by node $i = 100$: (a) DKF vs. KF (DKF is the smooth curve in red) and (b) Distributed Kalman filter estimate (in red) vs. the actual position of the object (in blue).



Distributed position estimation for a moving object by node $i = 25$: (a) DKF vs. KF (DKF is the smooth curve in red) and (b) Distributed Kalman filter estimate (in red) vs. the actual position of the object (in blue).