IFT 4030/7030, Machine Learning for Signal Processing

Week11: Graphs in Signal Processing and Machine Learning

Cem Subakan

Admin

- Mathieu a donné quelques feedbacks pour vos projets. Je vais essayer de faire la meme chose. Si vous avez des doutes, SVP prenez un rendez-vous avec moi / TAs!
  - Mathieu gave some feedback for the projects. I will try to do the same also. If you have doubts, please let me / TAs know!
- Le dernier lab est en vendredi. SVP soyez là!
  - The last lab is this friday. Please be there!

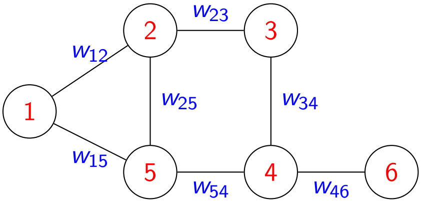
Today's Lecture

- Graphs!
- Using Graphs instead of vectors / matrices
  - On va utiliser des graphs au lieu de vecteurs / matrices
- Graph Signal Processing Basics
  - Les bases du traitement du signal avec des graphs
- Graph Machine Learning
  - De l'apprentissage automatique avec des graphs

Table of Contents

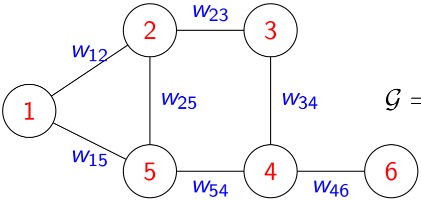
Graph

- A graph is a set of vertices (nodes) and edges.



Graph

- A graph is a set of vertices (nodes) and edges.

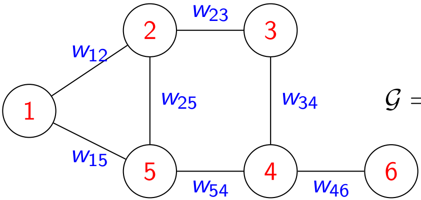


$$\mathcal{G} = \{V, E\}$$

- $\mathcal{G}$: Graph, V: Vertices (or Nodes), E : Edges

Graph

- A graph is a set of vertices (nodes) and edges.

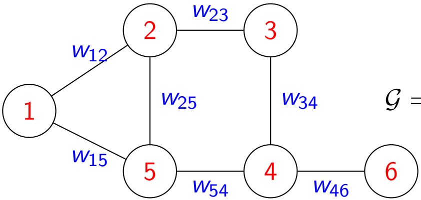


$\mathcal{G} = \{V, E\}$

- $\mathcal{G}$: Graph, V: Vertices (or Nodes), E : Edges
- Each node will have features / message. / Chaque node aura des features.

Graph

- A graph is a set of vertices (nodes) and edges.

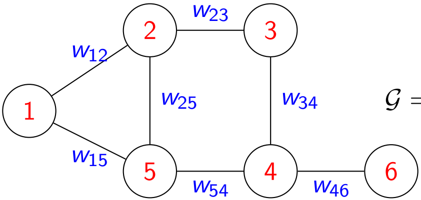


$\mathcal{G} = \{V, E\}$

- $\mathcal{G}$: Graph, V: Vertices (or Nodes), E : Edges
- Each node will have features / message. / Chaque node aura des features.
- Each edge has a weight / Chaque edge peut avoir un poids.

Graph

- A graph is a set of vertices (nodes) and edges.

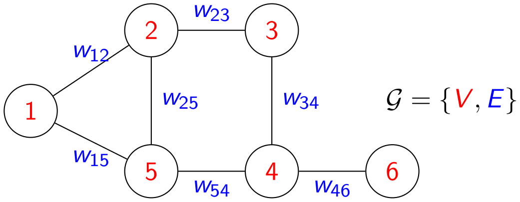


$\mathcal{G} = \{V, E\}$

- $\mathcal{G}$: Graph, V: Vertices (or Nodes), E : Edges
- Each node will have features / message. / Chaque node aura des features.
- Each edge has a weight / Chaque edge peut avoir un poids.
- Graphs might be directed or undirected. / Undirected: $1 \rightarrow 2 = 2 \rightarrow 1$
  - Les graph peut etre directionnel ou non.

Graph

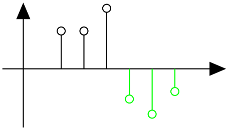
- A graph is a set of vertices (nodes) and edges.



- $\mathcal{G}$: Graph, V: Vertices (or Nodes), E : Edges
- Each node will have features / message. / Chaque node aura des features.
- Each edge has a weight / Chaque edge peut avoir un poids.
- Graphs might be directed or undirected. / Undirected: $1 \rightarrow 2 = 2 \rightarrow 1$
  - Les graph peut etre directionnel ou non.
- Graphs model **pairwise** relationships.
  - Les graphs modèle des relations **par paires**.

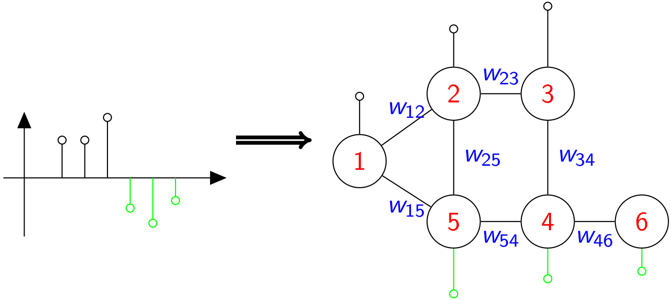
Representing a Signal as a Graph

- Associate each sample of the signal to a node. / On va associer chaque échantillon du signale avec un vertice.
  - Example: $x = [0.5, 0.5, 0.8, -0.4, -0.6, -0.3]$.



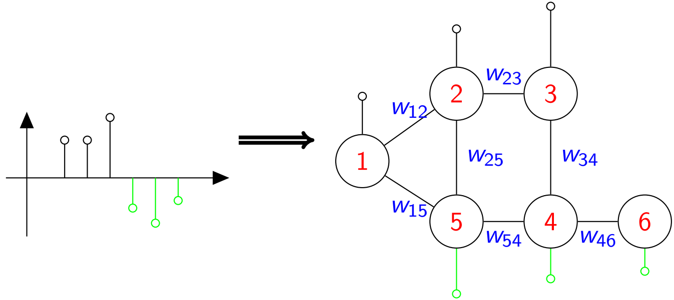
Representing a Signal as a Graph

- Associate each sample of the signal to a node. / On va associer chaque échantillon du signale avec un vertice.
  - Example: $x = [0.5, 0.5, 0.8, -0.4, -0.6, -0.3]$.



Representing a Signal as a Graph

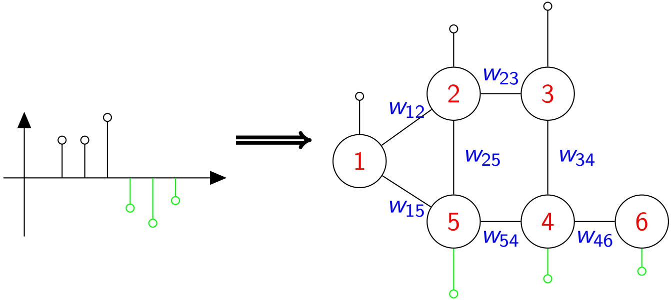
- Associate each sample of the signal to a node. / On va associer chaque échantillon du signale avec un vertice.
  - Example: $x = [0.5, 0.5, 0.8, -0.4, -0.6, -0.3]$.



- Signal values become node features. Why? What's the advantage? / Les valueurs du signal devient les features du node. C'est quoi l'avantage?

Representing a Signal as a Graph

- Associate each sample of the signal to a node. / On va associer chaque échantillon du signale avec un vertice.
  - Example: $x = [0.5, 0.5, 0.8, -0.4, -0.6, -0.3]$.



- Signal values become node features. Why? What's the advantage? / Les valueurs du signal devient les features du node. C'est quoi l'avantage?
- This enables us to explicitly model specific dependence structures, and it's a good representation if irregular signals.
  - Ça nous permet de modéliser des structures des dépendences spécifiques, et est une bonne manière de modéliser des signaux irregulaires.

How to represent a graph though?

- We understand that $\mathcal{G} = \{V, E\}$ defines a graphs.
  - On comprend que les ensembles V, E définissent un graph complétement. Mais comment est-ce qu'on répresente un graph numériquement?

How to represent a graph though?

- We understand that $\mathcal{G} = \{V, E\}$ defines a graphs.
  - On comprend que les ensembles V, E définissent un graph complétement. Mais comment est-ce qu'on répresente un graph numériquement?
- We will use an adjacency matrix A of size $N \times N$, for graph with N nodes.
  - On va utiliser une matrice d'adjacency du taille $N \times N$ pour un graphe de N nodes.

How to represent a graph though?

- We understand that $\mathcal{G} = \{V, E\}$ defines a graphs.
  - On comprend que les ensembles V, E définissent un graph complétement. Mais comment est-ce qu'on répresente un graph numériquement?
- We will use an adjacency matrix A of size $N \times N$, for graph with N nodes.
  - On va utiliser une matrice d'adjacency du taille $N \times N$ pour un graphe de N nodes.
- The element a_{ij} gives the weight of the edge ij. / a_{ij} donne le poids de l'edge ij.

How to represent a graph though?

- We understand that $\mathcal{G} = \{V, E\}$ defines a graphs.
  - On comprend que les ensembles V, E définissent un graph complétement. Mais comment est-ce qu'on répresente un graph numériquement?
- We will use an adjacency matrix A of size $N \times N$, for graph with N nodes.
  - On va utiliser une matrice d'adjacency du taille $N \times N$ pour un graphe de N nodes.
- The element a_{ij} gives the weight of the edge ij. / a_{ij} donne le poids de l'edge ij.
- Undirected graphs have symmetric adjacency matrices, such that $A^\top = A$.
  - Les graphes undirecteds ont des matrices adjacency symmetriques.

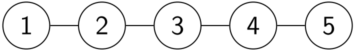
How to represent a graph though?

- We understand that $\mathcal{G} = \{V, E\}$ defines a graphs.
  - On comprend que les ensembles V, E définissent un graph complétement. Mais comment est-ce qu'on répresente un graph numériquement?
- We will use an adjacency matrix A of size $N \times N$, for graph with N nodes.
  - On va utiliser une matrice d'adjacency du taille $N \times N$ pour un graphe de N nodes.
- The element a_{ij} gives the weight of the edge ij. / a_{ij} donne le poids de l'edge ij.
- Undirected graphs have symmetric adjacency matrices, such that $A^\top = A$.
  - Les graphes undirecteds ont des matrices adjacency symmetriques.
- We will pack the node values in a vector $\mathrm{v} \in \mathbb{R}^N$. / On mets les features pour chaque node dans un vecteur v.
  - v_i gives the feature value of node i. v_i donne le valeur du feature du node i.

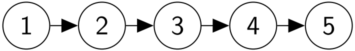
Table of Contents

Examples

- Undirected Path Graph

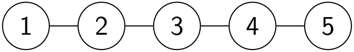


- Directed Path Graph



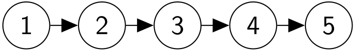
Examples

- Undirected Path Graph



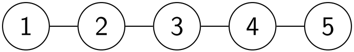
$$A = \begin{array}{cc} \textit{to} & \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} & \textit{from} \end{array}$$

- Directed Path Graph



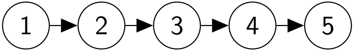
Examples

- Undirected Path Graph



$$A = \begin{array}{cc} \textit{to} & \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} & \textit{from} \end{array}$$

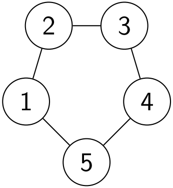
- Directed Path Graph



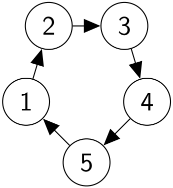
$$A = \begin{array}{cc} \textit{to} & \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} & \textit{from} \end{array}$$

Examples

- Undirected Ring Graph

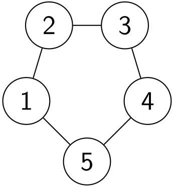


- Directed Ring Graph



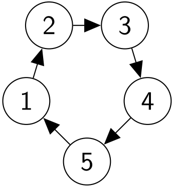
Examples

- Undirected Ring Graph



$$A = \begin{array}{cc} \textit{to} & \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix} & \textit{from} \end{array}$$

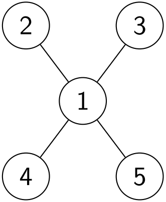
- Directed Ring Graph



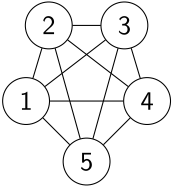
$$A = \begin{array}{cc} \textit{to} & \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} & \textit{from} \end{array}$$

More Examples

- Undirected Star Graph

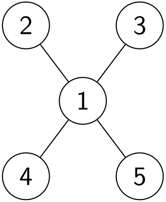


- Undirected Fully Connected Graph



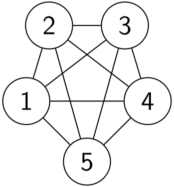
More Examples

- Undirected Star Graph



$$A = \begin{array}{c@{}l} \textit{to} & \\ \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} & \textit{from} \end{array}$$

- Undirected Fully Connected Graph



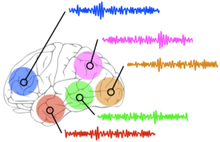
$$A = \begin{array}{c@{}l} \textit{to} & \\ \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix} & \textit{from} \end{array}$$

Real-World Examples

- Social Network Graph / Graphe d'un réseau sociale

Vertices: 1000 Twitter/X Users/Utilisateurs, **Edges:** Follower/not, **Node Features:** How many times did they tag IEEE MLSP 2023.

- Brain Measurements / Mésurements du cerveau



Vertices: Different regions , **Edges:** Structural Connectivity, **Node Features:** Brain activity measured as a 1d signal.

Table of Contents

Graph Signal Processing

- How do we work with such representations? Comment est-ce qu'on travaille avec tel représentations?
  - Graph Signal Processing
- Classical DSP assumes 'regular' data (It's assuming a graph). / DSP classique suppose une régularité dans les données.

Graph Signal Processing

- How do we work with such representations? Comment est-ce qu'on travaille avec tel représentations?
  - Graph Signal Processing
- Classical DSP assumes 'regular' data (It's assuming a graph). / DSP classique suppose une régularité dans les données.
- So, GSP is a generalization of DSP. / GSP est la généralization de DSP.
  - This helps us to model more interesting covariance structures. / Ça nous permet de modéliser des covariances plus intéressentes.

Graph Signal Processing

- How do we work with such representations? Comment est-ce qu'on travaille avec tel représentations?
  - Graph Signal Processing
- Classical DSP assumes 'regular' data (It's assuming a graph). / DSP classique suppose une régularité dans les données.
- So, GSP is a generalization of DSP. / GSP est la généralization de DSP.
  - This helps us to model more interesting covariance structures. / Ça nous permet de modéliser des covariances plus intéressentes.
- Also, GSP enables learning over graphs. / On peut aussi apprendre des modèles sur des graphes.

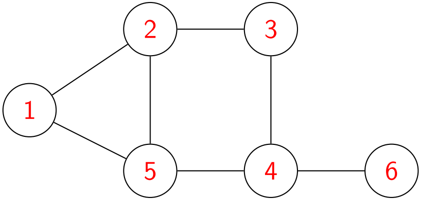
New things to worry about

- Classical Signal Processing
  - Time / Pixel Domain
  - We model relationships of samples laid on a regular grid / On modélise des relations sur un grille régulaire.
- Frequency Domain
  - Fourier Transform / DCT
- Graph Signal Processing
  - The graph domain.
  - Can model relationships described using an adjacency matrix. / On peut modéliser des rélations plus général défini par une matrice d'adjacency.
- Graph Spectral Domain
  - Graph Fourier Transform
- How do we move from spatial to graph domain? / Comment on va aller du domain spatial à la domaine de graphe?
  - We will use the Graph Laplacian L. / On va utiliser la graph laplacian L.

Table of Contents

The Graph Laplacian

- Consider this graph, with all weights equal to 1 / Considérons ce graph avec les poids égaux à 1.

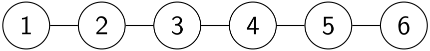


- $L := D - A$

$$\underset{L}{\overset{\text{Graph Laplacian}}{\begin{bmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}}} = \underset{D = \text{diag}(1^\top W)}{\overset{\text{Degree Matrix}}{\begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}} - \underset{A}{\overset{\text{Adjacency Matrix}}{\begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}}}$$

But, what is this graph Laplacian?

- Undirected Path Graph



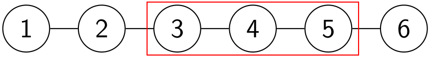
$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

- The Laplacian

$$\underbrace{\overbrace{\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}}^{\text{Graph Laplacian}}}_{L} = \underbrace{\overbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}^{\text{Degree Matrix}}}_{D = \mathrm{diag}(1^\top W)} - \underbrace{\overbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}}^{\text{Adjacency Matrix}}}_{A}$$

But, what is this graph Laplacian?

- Undirected Path Graph



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

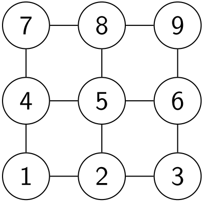
- The Laplacian

$$\underset{L}{\overset{\text{Graph Laplacian}}{\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}}} = \underset{D = \mathrm{diag}(1^\top W)}{\overset{\text{Degree Matrix}}{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}} - \underset{A}{\overset{\text{Adjacency Matrix}}{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}}}$$

This looks like the second derivative operator! (Discrete approximation to 2nd gradient) / C'est le noyau Laplacien, une approximation à la deuxième dérivée.

Grid

- Undirected Path Graph

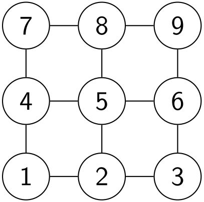


- The Laplacian

$$\underset{L}{\begin{bmatrix} 2 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 3 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 3 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 2 \end{bmatrix}} \underset{D = \mathrm{diag}(1^\top W)}{\begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}} \underset{A}{\begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}}$$

Grid

- Undirected Path Graph



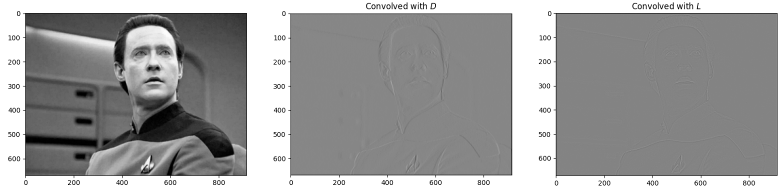
- The Laplacian

$$\underbrace{\begin{bmatrix} 2 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 3 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 3 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 2 \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}}_{D = \mathrm{diag}(1^\top W)} \underbrace{\begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}}_{A}$$

This looks like the second derivative operator! (Discrete approximation to 2nd gradient) / C'est le noyau Laplacien, une approximation à la deuxième dérivée.

Laplacian Operator on Images

Data is back!

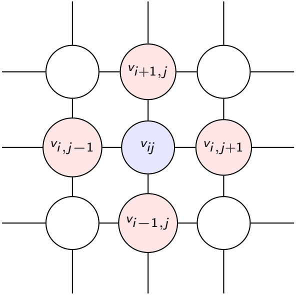


$$D = [1, -1] \quad L = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

L is an edge detector here.

The Graph Laplacian Properties

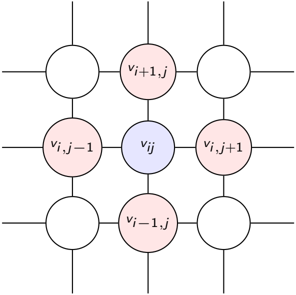
- We approximate the Laplace operator on the graph with Graph Laplacian / On approxime l'opérateur Laplace avec le Graph Laplacian.



- For undirected graphs L is symmetric. / Pour des graphs undirected L est symmetrique.
- Off diagonals are non-positive. / Les non-diagonales sont non-positives.
- Rows add up to zero. / Les lignes s'ajoutent à zéro.

The Graph Laplacian Properties

- We approximate the Laplace operator on the graph with Graph Laplacian / On approxime l'opérateur Laplace avec le Graph Laplacian.



- For undirected graphs L is symmetric. / Pour des graphs undirected L est symmetrique.
- Off diagonals are non-positive. / Les non-diagonales sont non-positives.
- Rows add up to zero. / Les lignes s'ajoutent à zéro.
- There's also the normalized Laplacian, $L_{\text{norm}} = D^{-1/2}(D - A)D^{-1/2}$, but we'll stick to L for now. / Il existe aussi la Laplacienne normalizé.

Table of Contents

The Graph Fourier Transform

- Given a Graph G containing a signal s, / Étant donné le graphe qui contient un signal s.
- Get Graph Laplacian L. / Obtiens la Laplancienne L.
- Do the eigenvalue decomposition on L. / Fait la décomposition éigen de L:
$$L = V \Lambda V^\top$$
- Transform the graph signal / Transforme le signal dans le graphe,
$$S = V^\top s$$

What does this mean?

- You can show that / On peut montrer que,

$$u^\top L u = \frac{1}{2}\sum_{i,j=1}^{N} \underbrace{w_{ij}}_{Adj.} \underbrace{(u_i - u_j)^2}_{\text{local variation}}$$

What does this mean?

- You can show that / On peut montrer que,

$$u^\top L u = \frac{1}{2}\sum_{i,j=1}^{N} \underbrace{w_{ij}}_{Adj.} \underbrace{(u_i - u_j)^2}_{\text{local variation}}$$

- What maximizes this measure of variation? / Que peut maximizer cette mésure de variation?

What does this mean?

- You can show that / On peut montrer que,

$$u^\top L u = \frac{1}{2} \sum_{i,j=1}^{N} \underbrace{w_{ij}}_{Adj.} \underbrace{(u_i - u_j)^2}_{\text{local variation}}$$

- What maximizes this measure of variation? / Que peut maximizer cette mésure de variation?
- $\max_u \; u^\top L u$ s.t. $u^\top u = 1$. What are these optimal u's? / Que sont ces u's optimales?

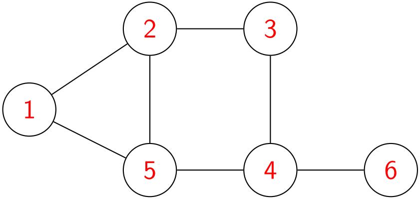
What does this mean?

- You can show that / On peut montrer que,

$$u^\top L u = \frac{1}{2} \sum_{i,j=1}^{N} \underbrace{w_{ij}}_{Adj.} \underbrace{(u_i - u_j)^2}_{\text{local variation}}$$

- What maximizes this measure of variation? / Que peut maximizer cette mésure de variation?
- $\max_u \; u^\top L u$ s.t. $u^\top u = 1$. What are these optimal u's? / Que sont ces u's optimales?
- Eigenvectors of u! / Vecteurs propres de u!
- Also, note that larger $u^\top L^\top u$ is, higher frequency we have in the signal u. / Notez que plus $u^\top L^\top u$ est large, plus le signale dans u est de haute fréquence.

Example



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

x	$x^\top Lx$
$[1, 1, 1, 1, 1, 1]$	0
$[0, 0, 1, 1, 0, 0]$	3
$[1, 0, 1, 0, 1, 0]$	5

Low and High Frequency Graph Signals

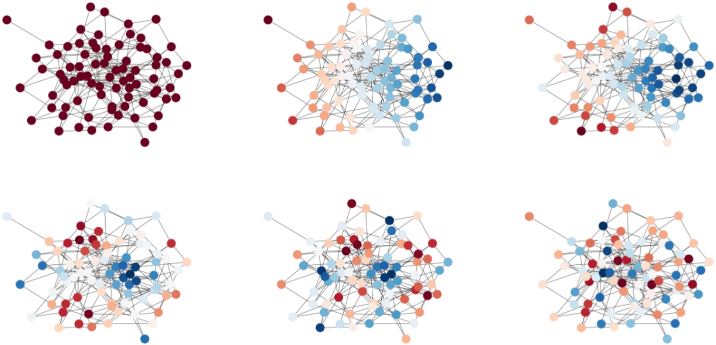
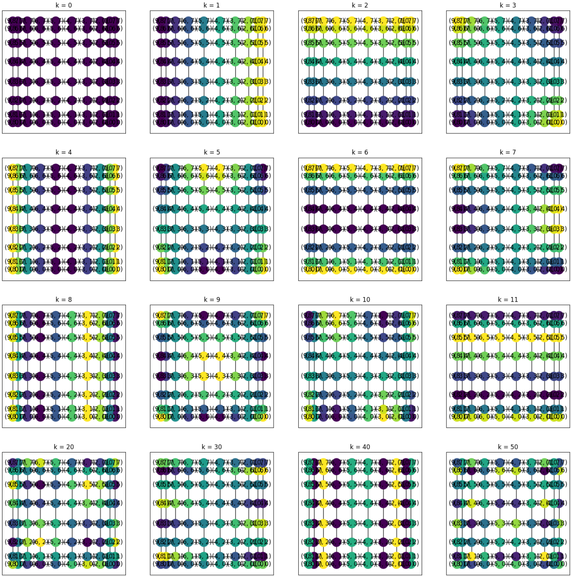
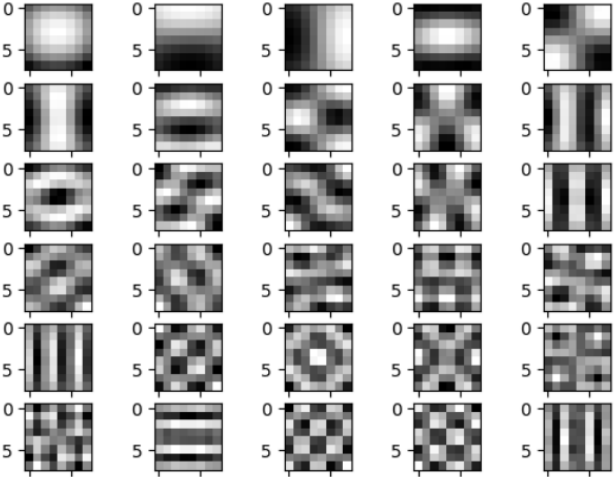


Image taken from UIUC MLSP class

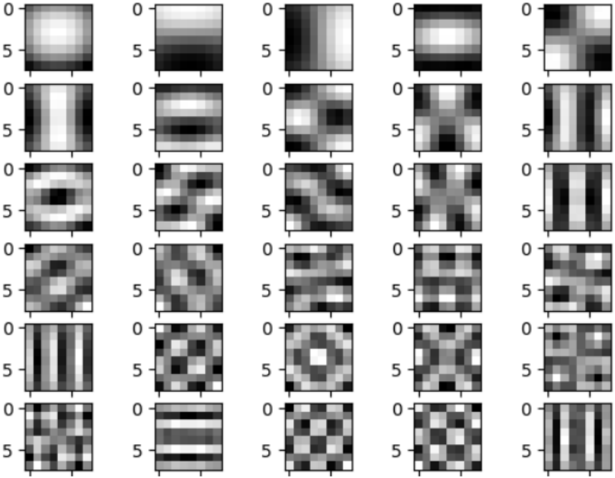
Grid Graph



Do you remember this?



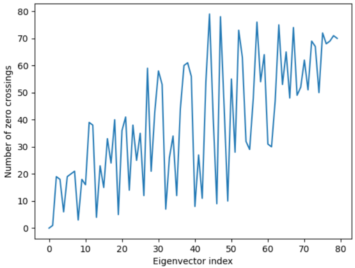
Do you remember this?



- It seems we were learning eigenvectors that correspond to a grid graph!
  - On apprenait des vecteurs propres qui correspondent à un graphe grille!

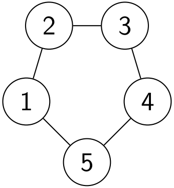
Eigenvalues as a measure of frequency

- The eigenvalues measure graph frequency
  - Les valeurs propres mesurent la fréquence de chaque base
- We can rank the eigenvectors by their eigenvalues / On peut mettre les vecteurs propres par leur valeurs propres
  - First eigenvectors are slow / change slowly. Les premiers vecteurs propres changent lentement
- The number of zero crossings for each eigenvector / le nombre de croisements zeros pour chaque vecteur propre:



Special Case: DFT

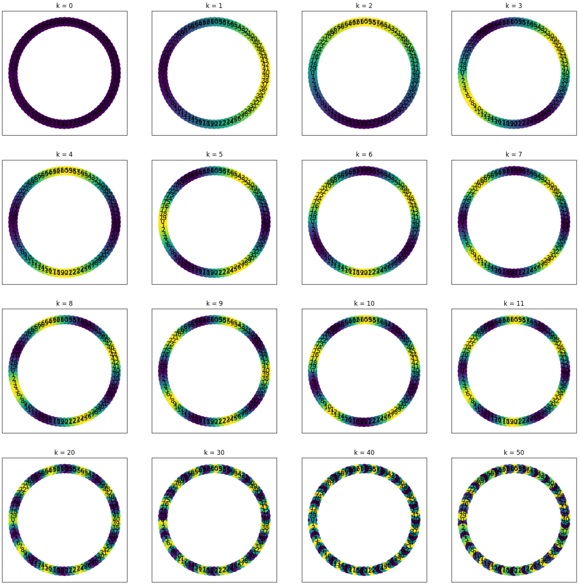
- Undirected Ring Graph



$$A = \begin{array}{cc} \textit{to} & \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix} & \textit{from} \end{array}$$

- Eigenvectors of L correspond to the Discrete Fourier Transform (DFT)! / Les vecteurs propres de L correspondent à DFT!

DFT



DFT Bases

- Paired Sine / Cosine Bases,

$$u_k(n) = \exp(j2\pi kn/N)$$

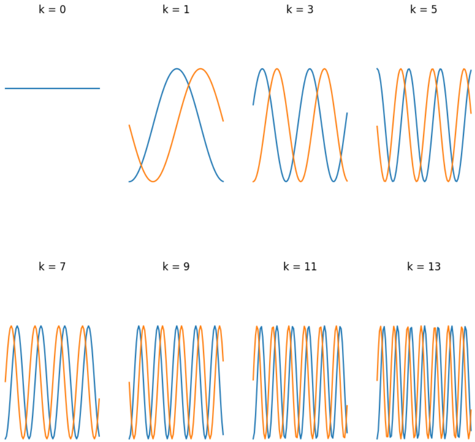
.

DFT Bases

- Paired Sine / Cosine Bases,

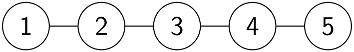
$$u_k(n) = \exp(j2\pi kn/N) = \cos(2\pi kn/N) + j\sin(2\pi kn/N)$$

.



Special Case 2, Discrete Cosine Transform

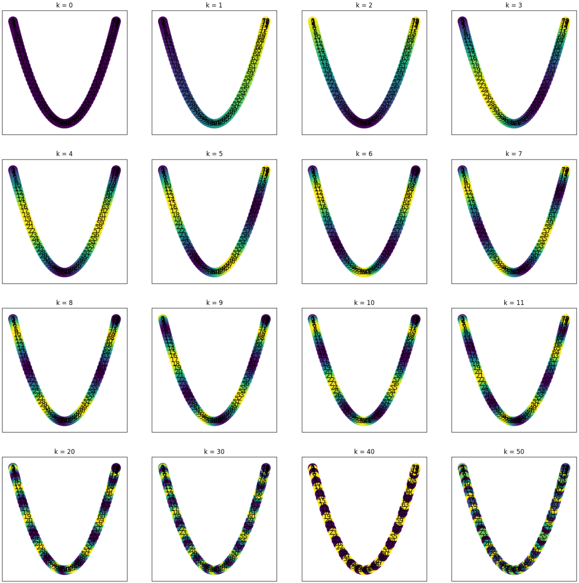
- Undirected Path Graph



$$A = \overset{\textit{to}}{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}} \textit{from}$$

- Eigenvectors of the Graph Laplacian will correspond to the DCT bases!
  - Les vecteurs propres du Laplacian correspondent aux bases DCT!

DCT



DCT-Type2 Bases

- Paired Sine / Cosine Bases,

$$u_{m,k}(n) \propto \cos((m + 0.5)k\pi/N)$$

.

DCT-Type2 Bases

- Paired Sine / Cosine Bases,

$$u_{m,k}(n) \propto \cos((m + 0.5)k\pi/N)$$

.

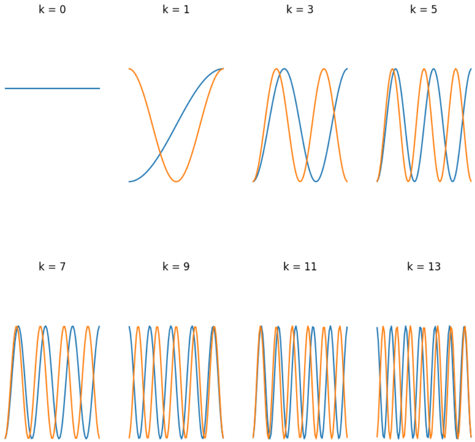


Table of Contents

Imposing a Graph

- Recap so far / Sommaire de ce qu'on a vu:
  - Graphs are collection of nodes and edges. / Un graphe est une collection des nodes et des edges.
  - A node is defined by an adjacency matrix A / Un graphe est défini par un adjacency matrix A.
  - Graph Laplacian $L = D - A$.
    - Is used to compute the Graph Fourier Transform. / Est utilisé pour calculer le Graph Fourier Transform.
    - This generalizes the classical Fourier Transform / Ça généralise le Fourier Transforme.
- We understand that standard SP assumes an underlying graph / Tout ça nous montre que le SP classique suppose un graphe.
  - Can we adjust the graph so that it's more suitable? / Peut-on adjuster le graph pour qu'il est plus ajusté?

Defining a Graph from a Signal

- We can define adjacency based on spatial location / On peut définir le voisinage en utilisant le localization spatiale, temporale

$$A_{ij}^{t} = \exp(-\alpha \| t_i - t_j \|)$$

- Or, we can define an adjacency based on sample values / Ou, on peut définir le voisinage en utilisant les valeurs
  - Do you remember this from somewhere else? / Souvenez-vous de ca d'ailleurs?

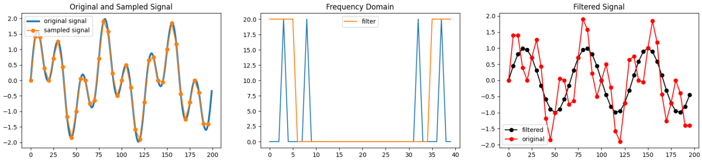
$$A_{ij}^{x} = \exp(-\beta \| x_i - x_j \|)$$

- We can define an overall adjacency matrix / On peut définir un matrice de voisinage ultime:

$$A = \gamma A^{t} + (1-\gamma) A^{x}$$

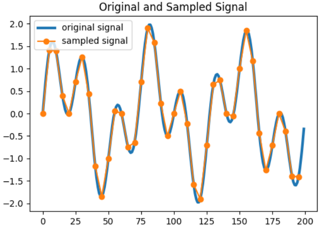
Classical filtering

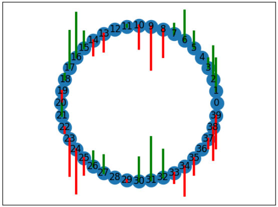
- $x_t = \cos(3\omega t) + \cos(8\omega t)$



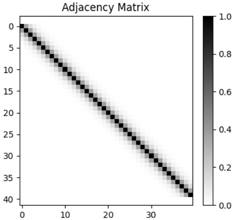
Let's do the same thing with Graphs

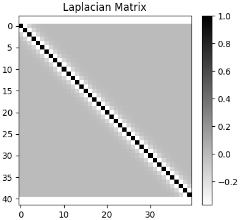
- Now we will define the same signal on a graph / On définit maintenant le meme signal sur un graphe





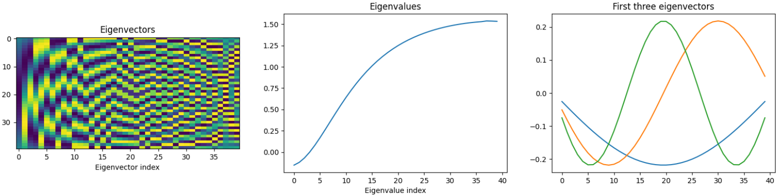
- Adjacency matrix, and the Laplacian (for a cycle graph)
  - $A_{ij} = \exp(-\alpha \| t_i - t_j \|)$





Step 1: Obtain the frequency bases of L

- $L = V \Lambda V^\top$



Graph Filtering Steps

- Transform onto the graph Laplacian eigenvectors.

$$X = \underbrace{V^\top}_{Eigenvec.} \underbrace{x}_{signal}$$

Graph Filtering Steps

- Transform onto the graph Laplacian eigenvectors.

$$X = \underbrace{V^\top}_{Eigenvec.} \underbrace{x}_{signal}$$

- Multiply with filter F

$$Z = F \odot X$$

Graph Filtering Steps

- Transform onto the graph Laplacian eigenvectors.

$$X = \underbrace{V^\top}_{Eigenvec.} \underbrace{x}_{signal}$$

- Multiply with filter F

$$Z = F \odot X$$

- Reconstruct

$$\widehat{x} = VZ$$

Graph Filtering Steps

- Transform onto the graph Laplacian eigenvectors.

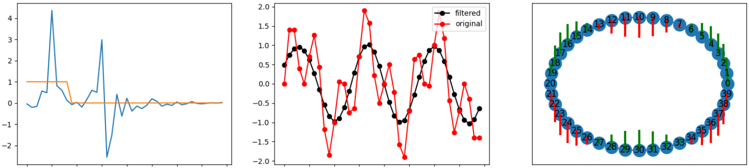
$$X = \underbrace{V^\top}_{Eigenvec.} \underbrace{x}_{signal}$$

- Multiply with filter F

$$Z = F \odot X$$

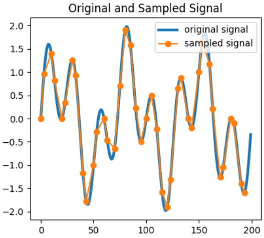
- Reconstruct

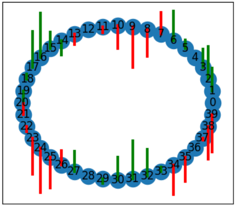
$$\widehat{x} = VZ$$



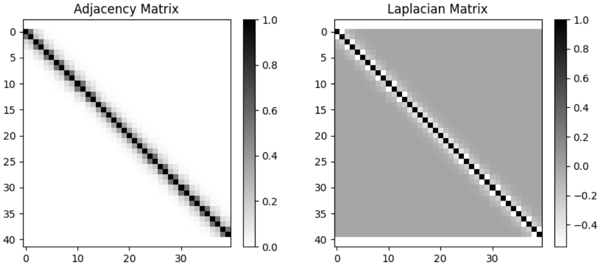
Nothing new with this, so why?

- What if the sample rate was not as regular / Et si le taux d'échantillonnage n'était pas si régulier?
- We can use a corresponding irregular graph maybe? / On peut utiliser la graph qui correspond?



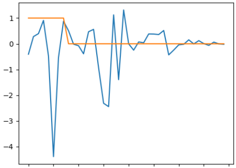


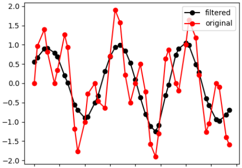
The new graph structure

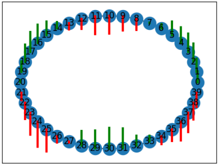


Non Uniform Graph Filtering

- We can still apply the same process as before / On peut encore appliquer le meme processus

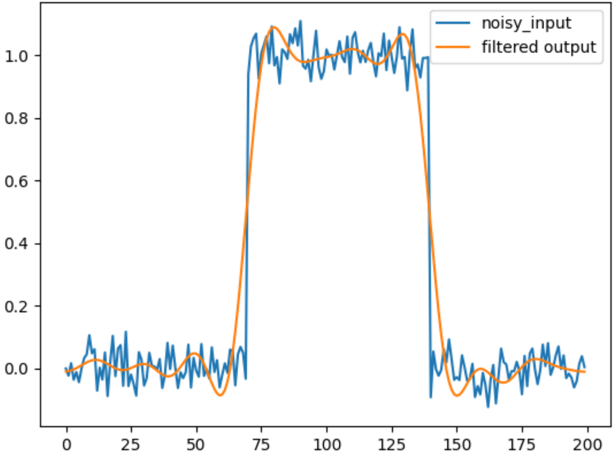






Encoding context from a signal

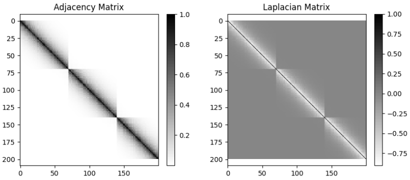
- Consider this noisy signal with steep edges / Considerons ce signal avec des sautes



Using the signal in the adjacency

- Can we somehow encode that closer values are similar? / Peut-on encoder que les valeurs qui sont proches sont plus similaires?
- Adjacency matrix:

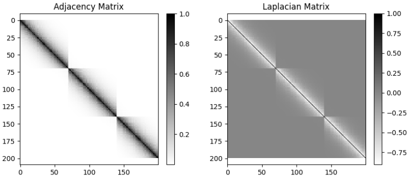
$$A_{ij} = \exp(-\alpha \|t_i - t_j\| - beta\|x_i - x_j\|)$$

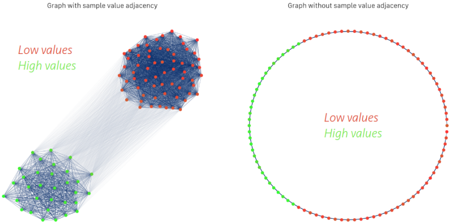


Using the signal in the adjacency

- Can we somehow encode that closer values are similar? / Peut-on encoder que les valeurs qui sont proches sont plus similaires?
- Adjacency matrix:

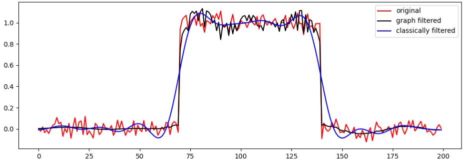
$$A_{ij} = \exp(-\alpha\|t_i - t_j\| - beta\|x_i - x_j\|)$$





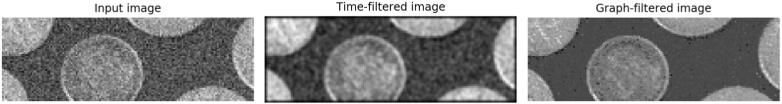
Edge preserving filter

- We can see that the graph filtering better preserves the edges / On voit que le filtre de graph mieux protège les sautes.



Edge preserving 2D example

- We can also apply the same thing on 2D signals / On peut appliquer la meme affaire sur des signaux 2D aussi.



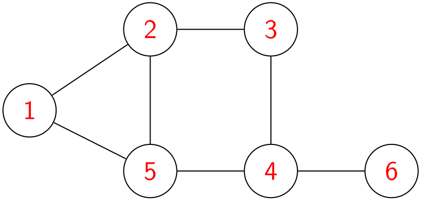
Defining Graphs from Data

- We already did this! / On a déjà fait ça
  - Eigendecomposition on an affinity matrix = KPCA/MDS/ISOMAP.
  - Clustering the eigenvectors = Spectral Clustering.

Table of Contents

Filtering in the graph spatial domain

- In regular signals we have the frequency domain / time domain filtering duality. / Dans les signaux réguliers on a une dualité entre filtrage dans le domaine de temps / domaine de fréquence.
- Filtering in the time domain is defined by convolution which uses shifts. Filtrage dans le domaine de temps utilise des shifts.
- We can also define a shift operator on graphs. / On define a shift operator over graphs



$$S = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} = I + A$$

Graph Filtering / Convolution

- Graph filtering definition: $h * x = \sum_k h_k S^k x$
- For instance for $h = [1, 1, 0.5]^\top$, $x = [-1, 2, 0, 0, 0, 0]^\top$

$$S = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} = I + A$$

$$Sx = [1, 1, 2, 0, 1, 0]^\top, \; S^2 x = [3, 5, 3, 3, 3, 0]^\top$$

$$y = h_0 x + h_1 S x + h_2 S^2 x = [1, 5, 2.5, 1.5, 2, 0]^\top$$

Graph Filtering / Convolution

- Graph filtering definition: $h * x = \sum_k h_k S^k x$
- For instance for $h = [1, 1, 0.5]^\top$, $x = [-1, 2, 0, 0, 0, 0]^\top$

$$S = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} = I + A$$

$$Sx = [1, 1, 2, 0, 1, 0]^\top,\ S^2x = [3, 5, 3, 3, 3, 0]^\top$$

$$y = h_0 x + h_1 Sx + h_2 S^2 x = [1, 5, 2.5, 1.5, 2, 0]^\top$$

- Notice that S defines a kernel with which we convolve! / Notez que S implique une sorte de noyau de convolution!

To summarize

- Signals can be defined as graphs / Signaux peut-etre défini comme des graphs.
  - It enables us to represent more structured sample relations / Ça nous permet d'établir des relations plus structurés entre les échantillons.
- We can define analogs to classical DSP / On peut définir des analogues aux DSP.
- Now, let's briefly mention that we can do ML also! / On peut faire du ML aussi!

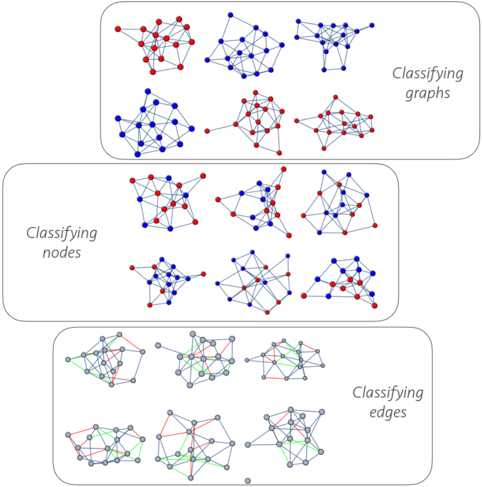
Table of Contents

Graph Neural Networks

- Neural Nets that operate on graphs. / Neural Nets qui fonctionne sur des graphs.
  - Using the graph shift matrix, we can define CNNs, RNNs.. / On peut définir des CNNs / RNNs en utilisant l'opérateur graph shift.
- Useful for things like / Utile pour des choses comme:
  - Molécules, social graphs, mesh representations, LIDAR, ...
- Hot topic / Je ne suis pas expert du tout!

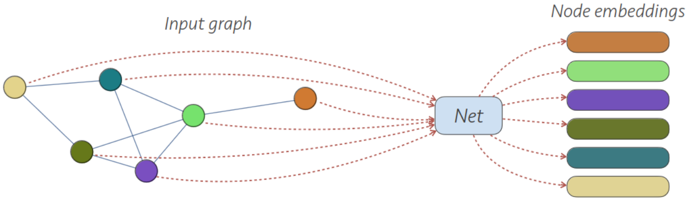
Types of Graph Level Processing

- Graph Level Processing, e.g. classifying a molecule
- Node Level Processing, e.g. LIDAR pixel classification
- Edge Level Processing e.g. Relationship classification in a social network



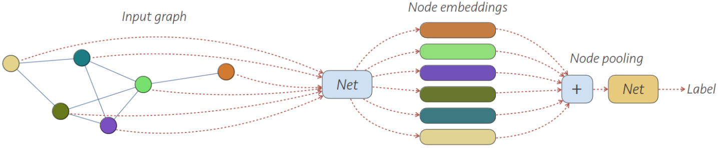
A very naive Graph Neural Network

- We can embed the nodes only / On embed les nodes seulement
  - Like you can do on your spectra in HW2-Q2 :)



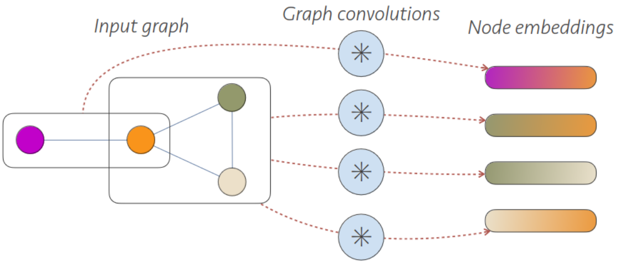
A very naive Graph Neural Network

- We can embed the nodes only / On embed les nodes seulement
  - Like you can do on your spectra in HW2-Q2 :)



Adding the graph structure

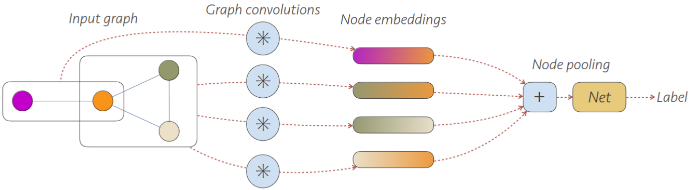
- We can also use graph convolutions to combine neighboring nodes when computing the node embeddings / On peut aussi inclure des convolutions graphes quand on calcule les node embeddings.
  - This enables us to model the graph structure / Ça nous permet de modéliser la structure de la graphe.



- Instead of using CNNs, it's possible to use RNNs / Attention Models, all that / Au lieu d'utiliser des CNNs, c'est possible d'utiliser des RNNs, modèles d'attention.

Adding the graph structure

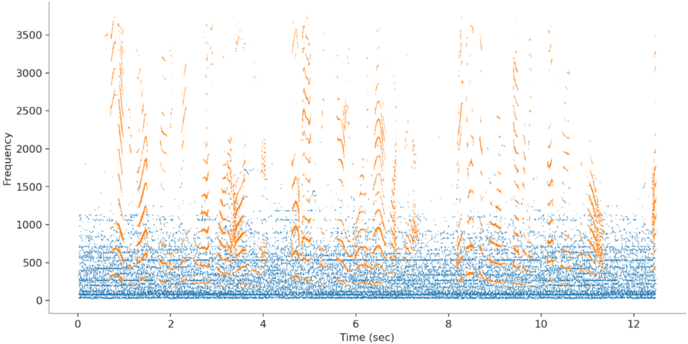
- We can also use graph convolutions to combine neighboring nodes when computing the node embeddings / On peut aussi inclure des convolutions graphes quand on calcule les node embeddings.
  - This enables us to model the graph structure / Ça nous permet de modéliser la structure de la graphe.



- Instead of using CNNs, it's possible to use RNNs / Attention Models, all that / Au lieu d'utiliser des CNNs, c'est possible d'utiliser des RNNs, modèles d'attention.

Audio Example

- Using point clouds for a spectrogram / Nuage de points pour un spectrogram (Look up re-assigned spectrograms)



- Instead of having series of vectors, we have a point cloud / on a un nuage de points au lieu de vecteurs: $C = [(f_1, t_1), (f_2, t_2), \dots, (f_N, t_N)]$.
- The paper (see the reading list) claims that training is faster, model is smaller, performance is better. (Best paper award in WASPAA 2021). / Le papier dit que l'entrainement est plus rapide, modèle est plus petit, et la performance est mieux.

Recap

- Signals are graphs! It's better to be aware of that to be able to generalize. / Les signaux sont des graphs en tout cas! Si on voit ça, on peut generaliser.
- Signal processing with Graph Perspective / Traitement du Signal avec un Perspective de Graphe
  - The Graph Fourier Transform
- Graph Neural Networks
  - Useful in many structured data domains (e.g. drug design) / Utile dans plusieurs données avec structure

Reading Material

- Graph Signal Processing:
  https://arxiv.org/pdf/1712.00468.pdf
- GNN tutorial:
  https://distill.pub/2021/understanding-gnns/
- Audio point cloud GNN stuff:
  https://arxiv.org/pdf/2105.02469.pdf

Next and last week

- I will talk about speech/audio. But let me know if you want me look into other things also. / Je vais parler sur speech/audio. Mais laissez-moi savoir si vous voulez voir d'autres sujets aussi.