Clustering Methods

So far... Supervised Learning

$X_1, X_2 \dots X_n$ inputs

$Y_1, Y_2 \dots Y_n$ outputs.

New input X: Predict Y.

Today: UNSUPERVISED LEARNING.

$X_1, X_2 \dots X_n$ inputs

No Outputs.

Goal: identify relevant subgroups, called clusters.

Example: images car / planes. No labels

identify: Cluster 1: car

Cluster 2: planes

① k-means

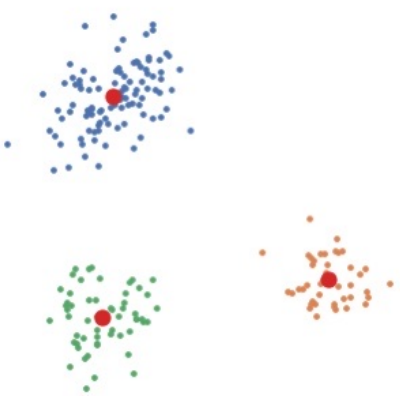
$$X = \{x_1 \dots x_n\} \subseteq \mathbb{R}^d$$

Goal of k-means:

Find a set of k representatives, called centroids.

$k = 3$

- centroids
- observations



Create the clusters by grouping points the closest to the same centroid.

⇝ How should the centroids be defined?

- $k = 1$: $\mathcal{X} = \{x_1, \dots, x_n\} \subseteq \mathbb{R}^d$

The best way to summarize $\mathcal{X}$ using one point:

→ Choose $y^* = \frac{x_1 + \dots + x_n}{n}$ the mean.

Actually, y^* is the minimizer of

$$F : y \in \mathbb{R}^d \longmapsto \frac{1}{n} \sum_{i=1}^{n} \|x_i - y\|^2.$$

proof : $\nabla F(y) = \frac{1}{n} \sum_{i=1}^{n} 2(y - x_i)$

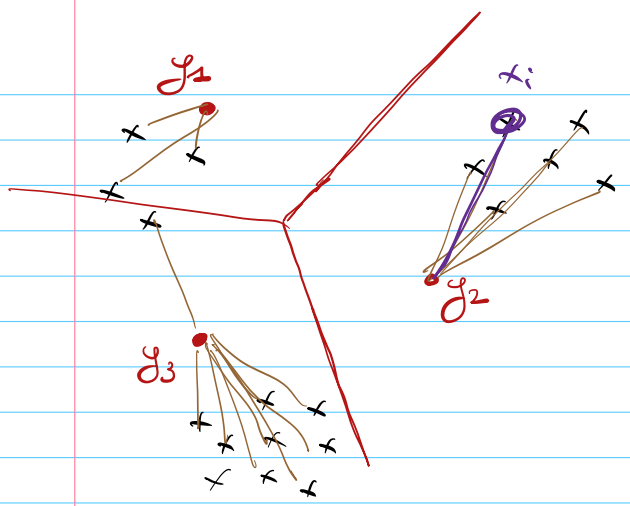
y such that $\nabla F(y) = 0 = 2\left(\frac{1}{n}\sum_{i=1}^{n} y - \frac{1}{n}\sum_{i=1}^{n} x_i\right)$

$\left(\frac{1}{n}\sum_{i=1}^{n} y = y\right)$

⇝ $y = y^* = \frac{1}{n} \sum_{i=1}^{n} x_i$.

⇝ For $k > 1$: Generalization

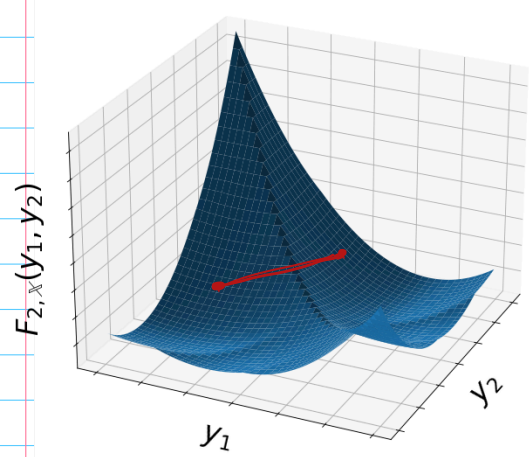
$$F_{k,\mathcal{X}} : (y_1, \dots, y_k) \in (\mathbb{R}^d)^n \longmapsto \frac{1}{n} \sum_{i=1}^{n} \min_{l=1\dots k} \|y_l - x_i\|^2$$



Choose (y_1, y_2, y_3) to minimize $\sum_{\quad} (\quad)^2$

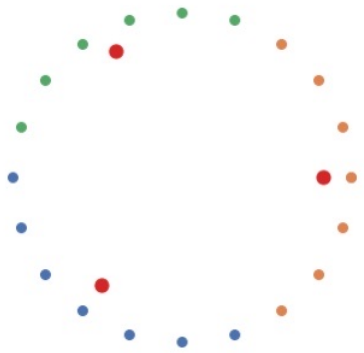
$\Rightarrow$ The k-means of X is the $\overset{a}{=}$ minimizer $(y_1^*, \dots, y_k^*)$ of $F_{k,X}$.

↳ also called centroids



The function $F_{k,X}$ is NOT convex. $(k \geq 2)$

$k=2$ $d=1$



Several k-means may exist.

⇝ How to compute the k-means?

LLOYD'S ALGORITHM

Initialization : centroids $y_1^0, \dots y_k^0$

For $t=0 \dots T-1$:

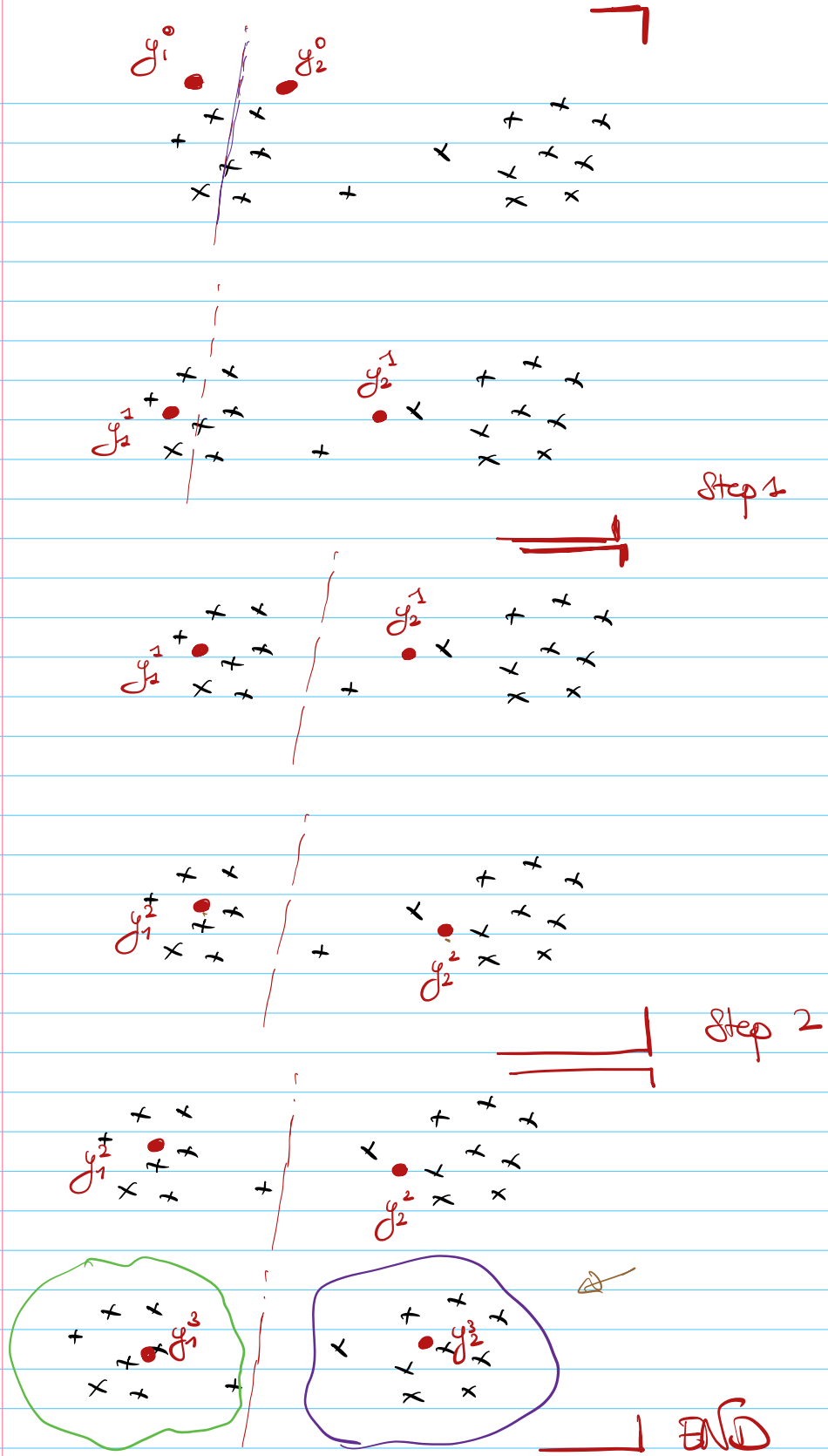
$\forall \ell = 1 \dots k$:

$I_\ell^t = \{ i : y_\ell^t \text{ is the closest to } x_i \}$ ← cluster ℓ at time t

size n_ℓ^t

$$y_\ell^{t+1} = \frac{1}{n_\ell^t} \sum_{i \in I_\ell^t} x_i$$

Output : $y_1^T \dots y_k^T$. ← Final centroids

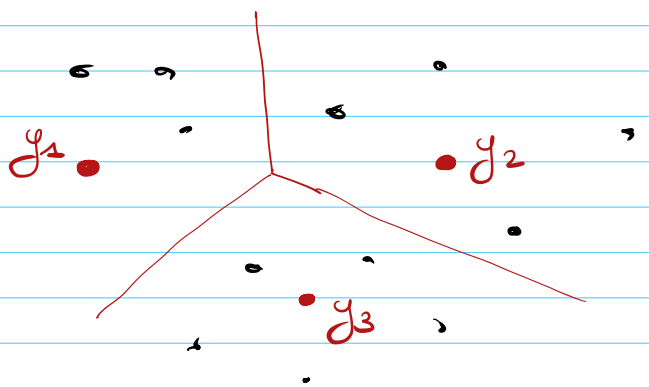

y_1^0
y_2^0
y_2^1
y_1^1
Step 1
y_1^1
y_2^1
y_1^2
y_2^2
Step 2
y_1^2
y_2^2
y_1^3
y_2^3
END

Prop: Lloyd's algorithm = Newton's method on $F_{k,\mathcal{X}}$.

Recall: Newton's method $F: \mathbb{R}^d \to \mathbb{R}$

$$\Rightarrow \quad y' = y - (\nabla^2 F(y))^{-1} \nabla F(y)$$

proof:



$$F_{k,\mathcal{X}}(y_1, \dots, y_k) = \frac{1}{n} \sum_{i=1}^{n} \min_{\ell=1\dots k} \|y_\ell - x_i\|^2$$

$$= \frac{1}{n} \sum_{\ell=1}^{k} \sum_{i \in I_\ell} \|x_i - y_\ell\|^2$$

$$\nabla F_{k,\mathcal{X}} = \begin{pmatrix} \frac{1}{2} d \\ 0 \\ \frac{1}{2} dk \end{pmatrix}$$

$$\nabla_{y_1} F_{k,\mathcal{X}}(\dots) = \frac{1}{n} \sum_{i \in I_1} 2(y_1 - x_i)$$

$$= \frac{2}{n} \left(n_1 y_1 - \sum_{i \in I_1} x_i \right)$$

$$= \frac{2n_1}{n} \left(y_1 - \underbrace{\frac{1}{n_1} \sum_{i \in I_1} x_i}_{\tilde{y}_1} \right)$$

$$\Rightarrow \quad \nabla F_{k,x}(-) = \begin{pmatrix} \nabla_{y_1} \\ \vdots \\ \nabla_{y_k} \end{pmatrix} = \frac{2}{n} \begin{pmatrix} n_1(y_1 - \tilde{y}_1) \\ \vdots \\ n_k(y_k - \tilde{y}_k) \end{pmatrix}$$

$$\Rightarrow \quad \nabla^2 F_{k,x}(-) = \begin{pmatrix} \nabla^2_{y_1 y_1} & \nabla^2_{y_1 y_2} & & \\ & \nabla^2_{y_2 y_2} & & \\ & & \ddots & \\ & & & \nabla^2_{y_k y_k} \end{pmatrix}$$

$$= \frac{2}{n} \begin{pmatrix} n_1 \mathrm{Id} & 0 & 0 \\ 0 & n_2 \mathrm{Id} & \\ 0 & & \ddots \\ & & & n_k \mathrm{Id} \end{pmatrix}$$

$$\text{w.o.} \quad \nabla^2 F_{k,x}^{-1}(\,) = \frac{n}{2} \begin{pmatrix} 1/n_1 & & 0 \\ & \ddots & \\ 0 & & 1/n_k \end{pmatrix}$$

$$\left(\nabla^2 F_{k,x}(\,)\right)^{-1} \nabla F_{k,x}(\,) = \begin{pmatrix} y_1 - \tilde{y}_1 \\ \vdots \\ y_k - \tilde{y}_k \end{pmatrix}$$

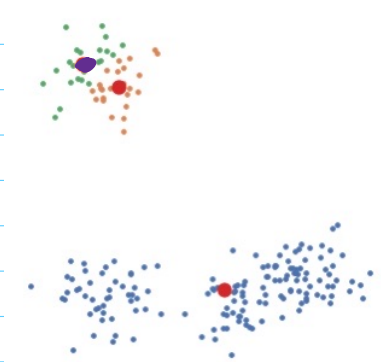
$\Rightarrow$ Newton's Method:

$$\begin{pmatrix} y_1' \\ \vdots \\ y_k' \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_k \end{pmatrix} - \begin{pmatrix} y_1 - \tilde{y}_1 \\ \vdots \\ y_k - \tilde{y}_k \end{pmatrix} = \begin{pmatrix} \tilde{y}_1 \\ \vdots \\ \tilde{y}_k \end{pmatrix} = \text{One step of Lloyd's Algorithm.}$$

■

⇒ Newton's method on a Non convex Function.

HW 8 ↳ May not converge with initialization too far away from optimum.



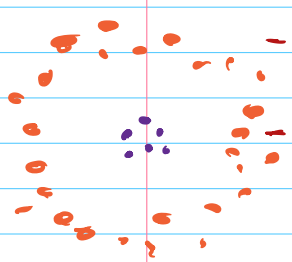
⇝ To Find a good initialization in practice: k-means ++.

↳ sklearn default method.

Summary: - most standard clustering algorithm

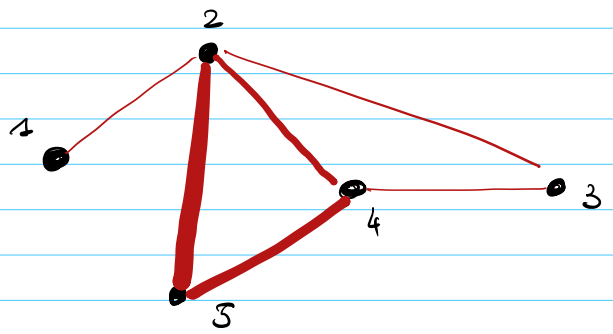
- will not work if clusters have "complex geometry"

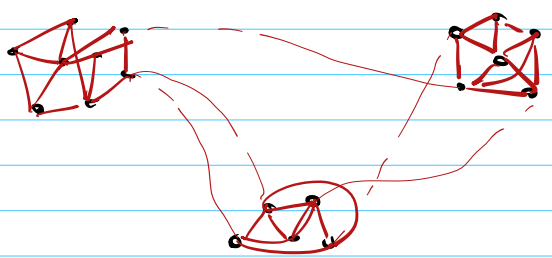
- Restricted to $x_i \in \mathbb{R}^d$



② Spectral Clustering

→ input = weighted graph representing similarities between observations.



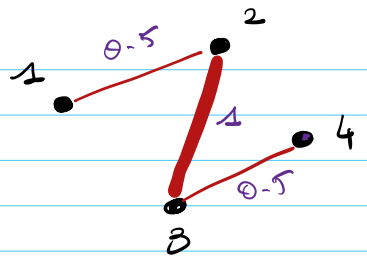


Three Clusters.

Def: A weighted graph $\mathcal{G}$ with

- n vertices
- weights $W = (W_{ij})_{i,j}$ $n \times n$ symmetric matrix

$W_{ij} \geq 0$

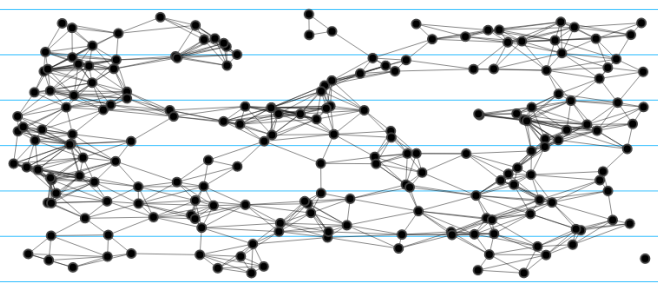


$$W = \begin{bmatrix} 1 & 0.5 & & \\ 0.5 & 1 & 1 & \\ & 1 & 1 & 0.5 \\ & & 0.5 & 1 \end{bmatrix}$$

Examples

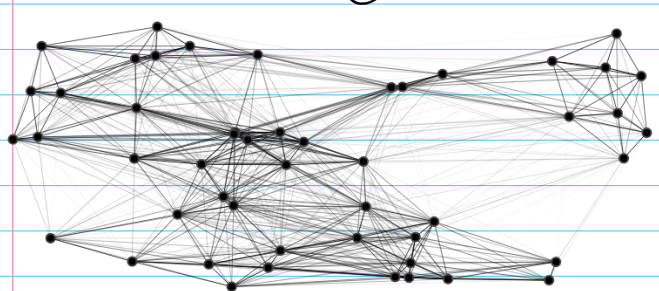
- (Non-weighted) graph: $\begin{cases} W_{ij} = 0 \text{ or } 1 \\ W_{ii} = 1 \end{cases}$

- ε-Neighborhood graph



$$W_{ij} = \begin{cases} 1 \text{ if } \|x_i - x_j\| \leq \varepsilon \\ 0 \text{ otherwise} \end{cases}$$

- Gaussian weights:

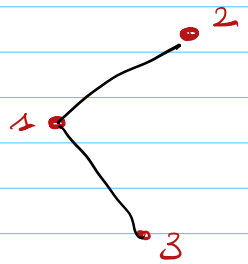


$$W_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

Def: $\mathcal{G}$ weighted graph

- **Neighbors** of i: j such that $W_{ij} > 0$

$\rightsquigarrow$ $i \sim_{\mathcal{G}} j$.

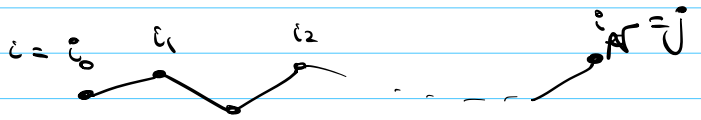


- **Degree** D_i of i:

$$D_i = \sum_{j=1}^{n} W_{ij}$$

$$D = \begin{pmatrix} D_1 & & 0 \\ & \ddots & \\ 0 & & D_n \end{pmatrix} = \text{degree matrix}$$

- The vertices i and j are **connected** if there is a path

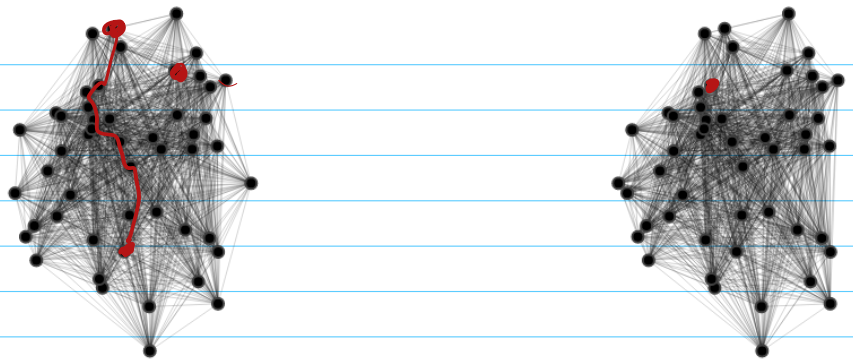


with $i_\ell \sim i_{\ell+1}$.

- A **connected component** is a set $\mathcal{C}$ of vertices such that

$\forall\, i, j \in \mathcal{C}$ i and j connected

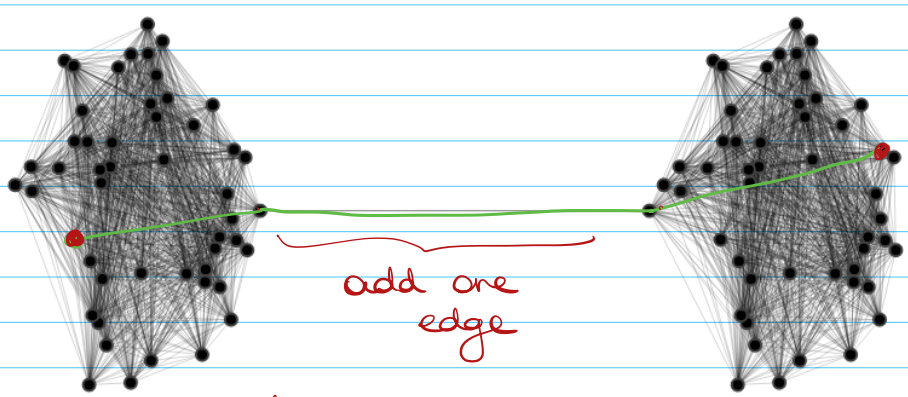
$\forall\, i \in \mathcal{C}, j \notin \mathcal{C}$ i and j NOT connected.



Two connected components

Idea 1 : Clusters = connected components.

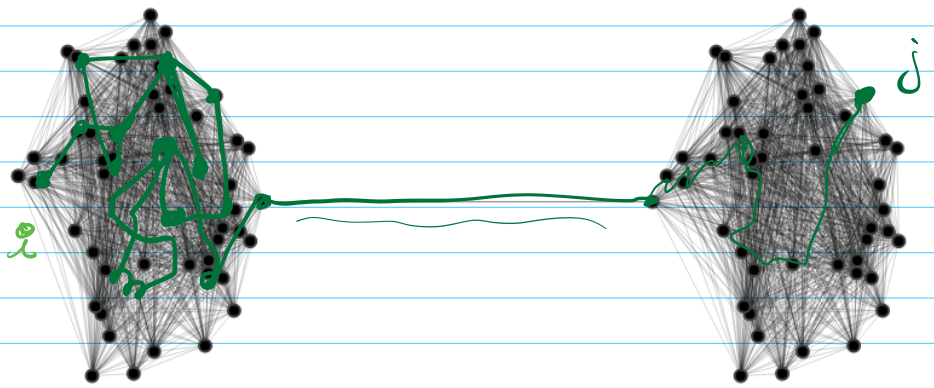
Problem: Not a robust Notion



→ only one connected component.

Idea 2: Make the notion of being connected quantitative.

RANDOM WALK ON THE GRAPH.

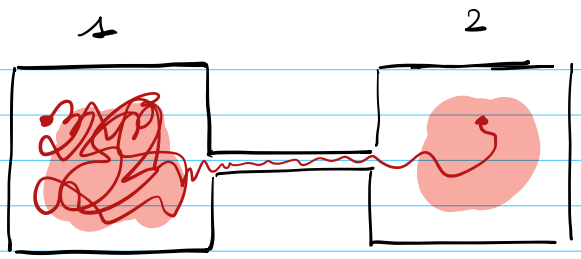


1. Start at i_0.
2. Select a random neighbour in of i_0.
3. Repeat.
4. Stop when j is reached.

$$i = i_0 \sim i_1 \sim i_2 \sim \cdots \sim i_K = j$$

if K is large : i and j are "almost" disconnected.

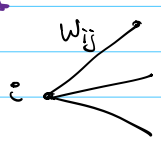
Analogy :



How much time does a molecule in a gas take to go from box 1 to box 2 ?

→ Probe to go from i to j in 1 step :

$$\sum_j Q_{ij} = \frac{1}{D_i} \sum_j W_{ij} = 1$$



$$Q_{ij} = \frac{1}{D_i} W_{ij}$$

↳ so that $\sum_j Q_{ij} = 1$

$Q = (Q_{ij})$ is the probability transition matrix of the walk.

$$\begin{pmatrix} 1-Q_{11} & -Q_{12} & -Q_{13} \\ & 1-Q_{22} & \end{pmatrix} \begin{pmatrix} \\ Q_{ij} \\ \end{pmatrix}$$

→ $L = I_n - Q$ = Laplacian of $\mathcal{G}$.

⇒ Spectral properties of L contain information about the geometry of $\mathcal{G}$.

elements of A

$A \subseteq \{1, \dots, n\}$ $\quad e_A = (0 \dots 0\,1\,0 \dots 0\,1 \dots) \in \mathbb{R}^n$

$$(e_A)_i = \begin{cases} 1 & \text{if } i \in A \\ 0 & \text{otherwise.} \end{cases}$$

$A = \{1\}$ $\quad e_A = (1\,0 \dots)$

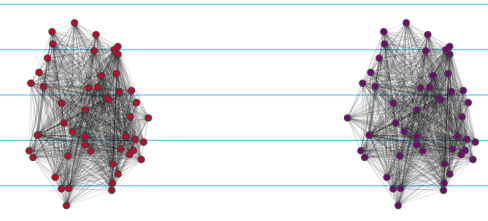
$A = \{1,2\}$ $\quad e_A = (1\,1\,0 \dots)$

Prop :

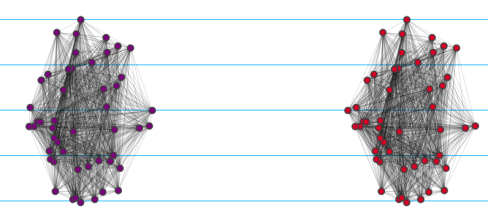
1. L has nonnegative eigenvalues.
2. 0 is an eigenvalue of L.
3. The multiplicity of 0 is the number k of connected components of G.

A basis of the eigenspace is $\{e_{C_1}, \dots, e_{C_k}\}$

↳ connected component



$e_{C_1} = \begin{pmatrix} 1 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$



$e_{C_2} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$

proof : ⚠ L is NOT symmetric !

"symmetric Laplacian" $L' = D^{-1/2} L D^{-1/2} = I_n - D^{-1/2} D^{-1} W D^{-1/2}$

$= I_n - D^{-1/2} W D^{-1/2}$ ← is symmetric

$Lu = \lambda u \iff L'v = \lambda v \qquad v = D^{1/2} u$

⇒ L and L' have the same eigenvalues.

Let $v \in \mathbb{R}^n$

$v^T L' v =$

$$= \frac{1}{2} \sum_{1 \le i,j \le n} w_{ij} \left(\frac{v_i}{\sqrt{D_i}} - \frac{v_j}{\sqrt{D_j}} \right)^2 \geq 0.$$

(2) Take $v = (\sqrt{D_1} \dots \sqrt{D_n})$ ⇒ $v^T L' v = 0$

⇒ 0 eigenvalue of L'

⇒ 0 eigenvalue of L.

(3) v eigenvector of L' $\quad [v = D^{1/2}u]$

$\Leftrightarrow$ u eigenvector of L.

$\Rightarrow$ When do we have

$$0 = v^T L' v = \frac{1}{2} \sum_{1 \le i,j \le n} w_{ij} \left(\frac{v_i}{\sqrt{D_i}} - \frac{v_j}{\sqrt{D_j}} \right)^2$$

$$= \frac{1}{2} \sum_{1 \le i,j \le n} w_{ij} \left(u_i - u_j \right)^2 \Rightarrow$$ $u_i = u_j$ if $w_{ij} > 0$

$k = 1$: one connected component.

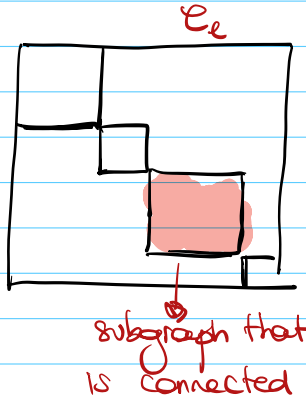
- $u = (1 \ \dots \ 1)$ is an eigenvector.
- if $u \in \mathbb{R}^n$ s.t. $u^T L' u = 0$

Path $\quad i = i_0 \sim i_1 \sim \dots \sim i_k = j$

$u_i = u_{i_1} = u_{i_2} = \dots = u_{i_k} = u_j$

$\Rightarrow$ $u = (c, c, \dots, c)$

$k > 1$:



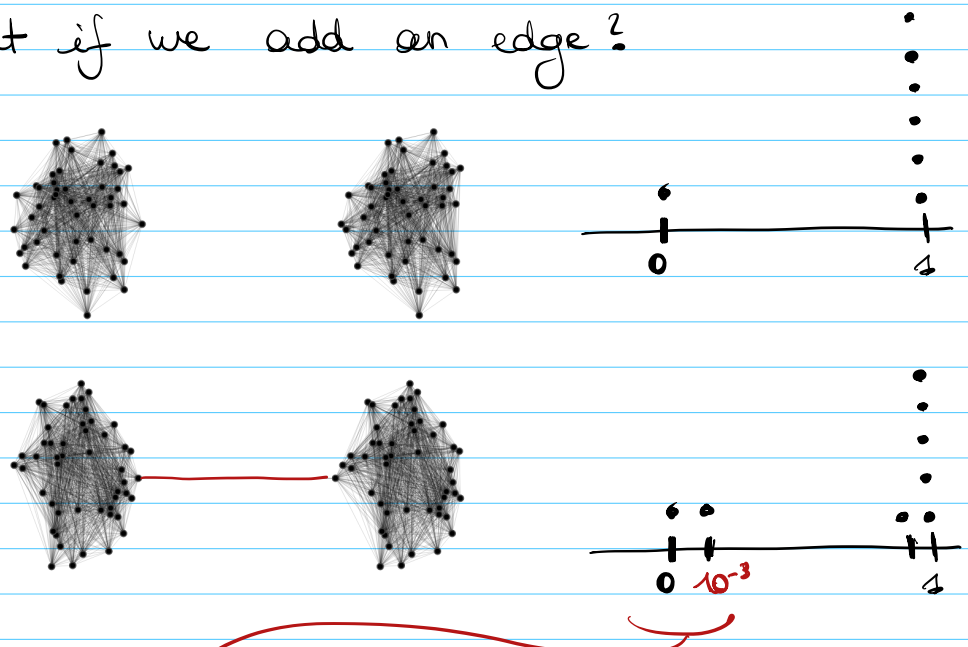
$\Rightarrow$ e_{C_ℓ} is an eigenvector.

$\blacksquare$

Summary:

Multiplicity of 0	Associated Eigenvectors
⇕	⇕
Number of connected components	Connected components

What if we add an edge?



Stability of the spectrum with respect to small perturbations.

k clusters = k small eigenvalues

↳ associated eigenvectors give the clusters.

SPECTRAL CLUSTERING

1- Compute L

2- Compute the k first eigenvalues / eigenvectors of L.

$0 = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_k$

$u_1 \quad u_2 \quad \dots \quad u_k$ eigenvectors. $(\in \mathbb{R}^n)$

3- Let $x_i = (\langle u_1, e_i \rangle, \dots, \langle u_k, e_i \rangle) \in \mathbb{R}^k$

$= (0 \dots 1 \dots 0)$

= represents i.

4- Apply k-means on $(x_1, \dots x_n)$.

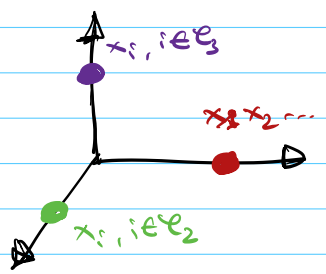
⇒ if k connected components:

$$u_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad u_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \quad u_3 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$i \in \mathcal{C}_1 \quad x_i = (1, 0, 0)$

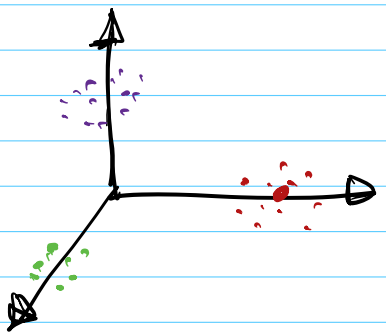
$i \in \mathcal{C}_2 \quad x_i = (0, 1, 0)$

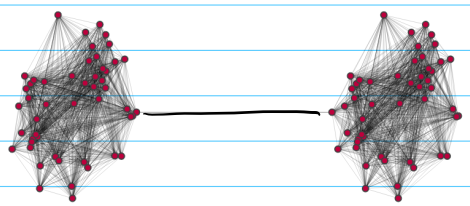
$i \in \mathcal{C}_3 \quad x_i = (0, 0, 1)$



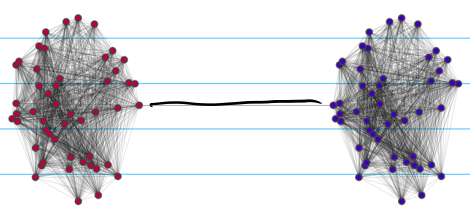
⇒ With "approximate" connected components:

clusters





$d = 0$



$d = 10^{-3}$