



# SPECTRAL CLUSTERING FROM A GEOMETRICAL VIEWPOINT

Tyrus Berry, Timothy Sauer



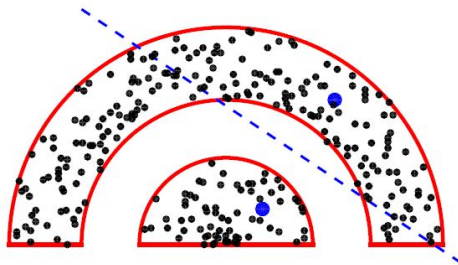
# Outline of the presentation

- Understanding the classical algorithm for spectral clustering
- 3 limits of the classical method and how to overcome them
- Conclusion

# Understanding spectral clustering

## Motivations:

- K-means limitations:



(a)

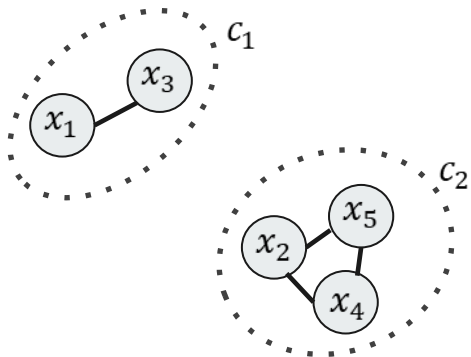
- Clustering definition: “The intuition of clustering is to separate points in different groups according to their similarities”

# Understanding spectral clustering

## Finding connected components in the perfect case:

Assumption: The data is already given with the knowledge of pairwise similarity

→ Number of clusters = number of connected components in the graph (« *perfect case* »)



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

5 points in the plane with their similarity links form 2 clusters

Corresponding adjacency matrix

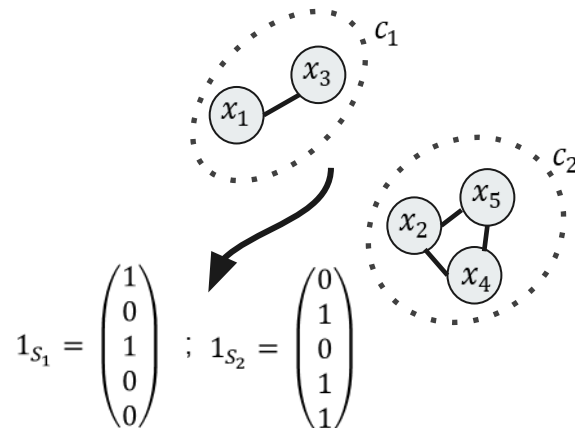
# Understanding spectral clustering

## Indicator vectors

Given a subset of vertices  $A \subset V$  we define:

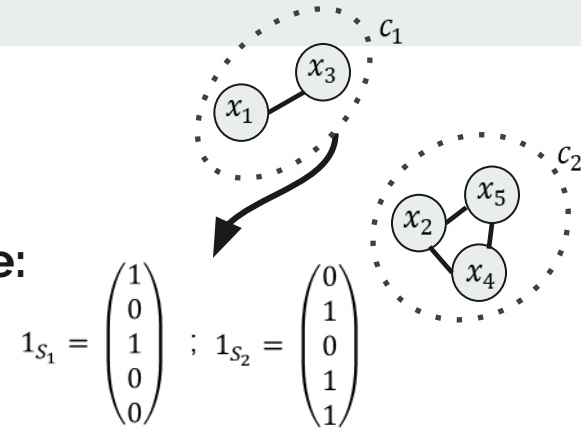
$$\mathbf{1}_A = (f_1, \dots, f_n)^\top \quad \text{where} \quad f_i = \begin{cases} 1 & \text{if } i \in A \\ 0 & \text{otherwise} \end{cases}$$

(indicator vector)



# Understanding spectral clustering

Finding connected components in the perfect case:



**Definition:** • Graph Laplacian  $L = D - W$  with  $D = \text{diag}(W1)$  ;  
1 the vector of ones ;  
 $W$  the adjacency matrix

**Theorem:** The multiplicity  $k$  of the eigenvalue 0 of the graph Laplacian  $L$  is equal to the number of  $W$ - connected components. The eigenspace of the eigenvalue 0 is spanned by the indicator vectors of those connected components.

**Note:** There are other possible definitions of  $L$  , we will stick to the simplest one

# Understanding spectral clustering

## Finding connected components in the perfect case:

Outline of the Proof: - With  $k=1$ , we assume that  $\varphi$  is an eigenvector of 0

$$0 = \varphi^\top \mathbf{L} \varphi = \sum_{i,j=1}^n w_{ij} (\varphi_i - \varphi_j)^2 \Rightarrow (\forall i, j : w_{ij} \neq 0), \varphi_i = \varphi_j$$

Then, the eigenspace of the eigenvalue 0 of  $\mathbf{L}$  is spanned by the vector  $\mathbf{1}$

- For any  $k$ , if we name  $\mathbf{L}_i$  the graph Laplacian of the  $i$ -th component, we have:

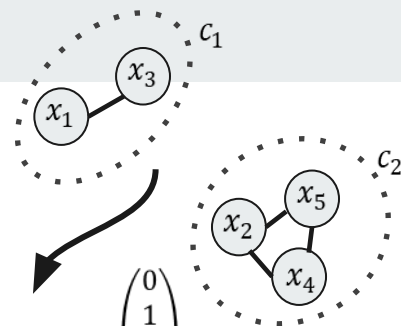
$$\mathbf{L} \sim \begin{pmatrix} \mathbf{L}_1 & & & \\ & \mathbf{L}_2 & & \\ & & \ddots & \\ & & & \mathbf{L}_k \end{pmatrix} \Rightarrow \{\lambda\}_L = \bigcup_{i=1}^k \{\lambda\}_{L_i}$$

Then, the eigenspace of the eigenvalue 0 of  $\mathbf{L}$  is spanned by the vectors  $\mathbf{1}_{S_i}$

# Understanding spectral clustering

Finding connected components in the perfect case:

$$\mathbf{1}_{S_1} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} ; \mathbf{1}_{S_2} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$



Implications: In our example, let's take  $\varphi_1$  and  $\varphi_2$  the 2 orthogonal eigenvectors for  $L$  ( $L$  is symmetric, positive semi-definite).

Then:

$$\varphi_1 = \sum_{i=1}^k \alpha_i^{(1)} \mathbf{1}_{S_i} = \begin{pmatrix} \alpha_1^{(1)} \\ \alpha_2^{(1)} \\ \alpha_1^{(1)} \\ \alpha_2^{(1)} \\ \alpha_2^{(1)} \end{pmatrix} ; \varphi_2 = \sum_{i=1}^k \alpha_i^{(2)} \mathbf{1}_{S_i} = \begin{pmatrix} \alpha_1^{(2)} \\ \alpha_2^{(2)} \\ \alpha_1^{(2)} \\ \alpha_2^{(2)} \\ \alpha_2^{(2)} \end{pmatrix}$$

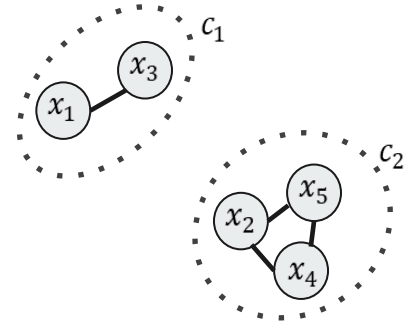
And if we concatenate the two vectors in the same matrix  $\phi$ :

$$\phi = \begin{pmatrix} \alpha_1^{(1)} & \alpha_1^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \\ \alpha_1^{(1)} & \alpha_1^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \end{pmatrix}$$



# Understanding spectral clustering

Finding connected components in the perfect case:



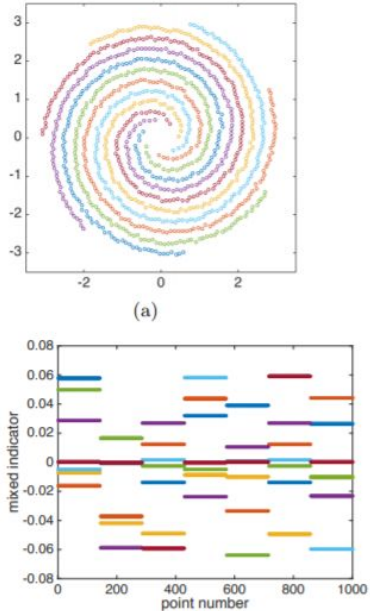
Implications:

Now, if we take the rows of  $\phi$ , we end up with 6 new vectors:

$$\phi = \begin{pmatrix} \alpha_1^{(1)} & \alpha_1^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \\ \alpha_1^{(1)} & \alpha_1^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \\ \alpha_1^{(1)} & \alpha_1^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \end{pmatrix} \Rightarrow y_1 = \begin{pmatrix} \alpha_1^{(1)} \\ \alpha_2^{(2)} \end{pmatrix}, y_2 = \begin{pmatrix} \alpha_2^{(1)} \\ \alpha_2^{(2)} \end{pmatrix}, y_3 = \begin{pmatrix} \alpha_1^{(1)} \\ \alpha_1^{(2)} \end{pmatrix}, y_4 = \begin{pmatrix} \alpha_2^{(1)} \\ \alpha_2^{(2)} \end{pmatrix}, y_5 = \begin{pmatrix} \alpha_2^{(1)} \\ \alpha_2^{(2)} \end{pmatrix}$$

In this new space, we have:  $y_1 = y_3$  ;  $y_2 = y_4 = y_5$  ... which does look like our clusters!

Run a k-mean (with k=2) algorithm on the  $y_i \Rightarrow$  Identification of the clusters



# Understanding spectral clustering



## What do we do in reality ?

### Algorithm:

INPUT: Cloud of  $N$  points

- Find the similarities, **build** the adjacency matrix  $W$  and **compute**  $L$
- **Compute** the  $k$  eigenvectors of  $L$  for the eigenvalue 0
- **Concatenate** them in a matrix  $\phi$
- **Run** the  $k$ -means algorithm in the  $N$  rows of  $\phi$
- **Return** the labels of the clusters for each point

### Questions:

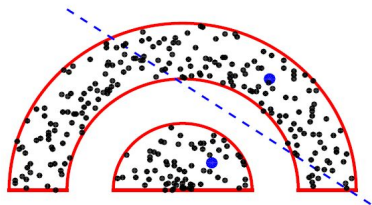
- *How do we define the notion of similarity ? How do we build a good adjacency matrix ?*
- *What if the graph we build is not « perfect »? The method still works ? Why ?*

# Understanding spectral clustering

## Defining a good notion of similarity

- Assumption:  $N$  data points are sampled from a manifold with  $k$  connected components
- Goal: Recover the manifold by building a graph and **identify** the connected components
- How? Define the notion of similarity between pairs of points  $(x_i, x_j)$  with a kernel like:

$$W_{ij} = h\left(\frac{\|x_i - x_j\|^2}{\epsilon^2}\right), \quad \text{where } h(x) = \begin{cases} 1 & \text{if } x < 1 \\ 0 & \text{otherwise.} \end{cases}$$



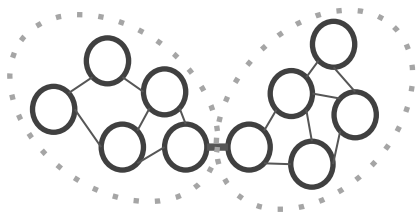
(a)

$$\begin{aligned} \epsilon \rightarrow 0 &\Rightarrow k = N \\ \epsilon \rightarrow +\infty &\Rightarrow k = 1 \end{aligned}$$

# Understanding spectral clustering

Why does it work in practice ?

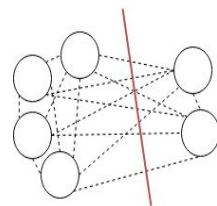
Connected clusters



Gaussian Kernel

$$h(x) = e^{-x/4} \Rightarrow W \text{ has no } 0 \\ \Rightarrow \text{only 1 cluster ?}$$

- **Graph cut:** This spectral method is used to solve a relaxed problem of graph cut
- **Perturbation theory:** The matrices we are working with are not *too far* from the ideal ones



$\Rightarrow$  It still works

# Limits of the method and how to overcome them

- Limits:
- Necessity to use another clustering method in the last step
  - Necessity to tune an hyperparameter  $\epsilon$  to find a good number of clusters
  - Method doesn't cope with different density of points within the clusters ( $\epsilon$  is global)

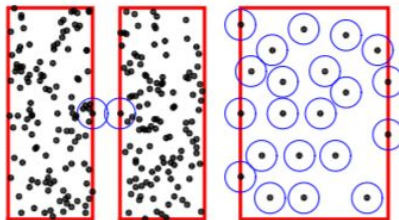


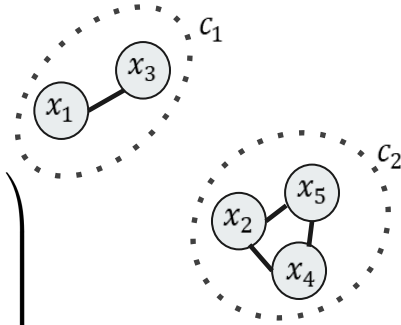
FIG. 3.1. An example with varying densities. Any spectral method that relies on a single global bandwidth (denoted by the circles) cannot properly divide the example into three clusters.

# Limits of the method and how to overcome them

## Avoiding the use of an other clustering algorithm

- What we have access to:

$$\varphi_1 = \sum_{i=1}^k \alpha_i^{(1)} \mathbf{1}_{S_i} = \begin{pmatrix} \alpha_1^{(1)} \\ \alpha_2^{(1)} \\ \alpha_1^{(1)} \\ \alpha_2^{(1)} \\ \alpha_2^{(1)} \end{pmatrix} ; \varphi_2 = \sum_{i=1}^k \alpha_i^{(2)} \mathbf{1}_{S_i} = \begin{pmatrix} \alpha_1^{(2)} \\ \alpha_2^{(2)} \\ \alpha_1^{(2)} \\ \alpha_2^{(2)} \\ \alpha_2^{(2)} \end{pmatrix} \quad \phi = \begin{pmatrix} \alpha_1^{(1)} & \alpha_1^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \\ \alpha_1^{(1)} & \alpha_1^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \end{pmatrix}$$



- What we want to have: A matrix **C** containing, for each data point (i.e row), the one-hot encoding of its cluster

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

# Limits of the method and how to overcome them

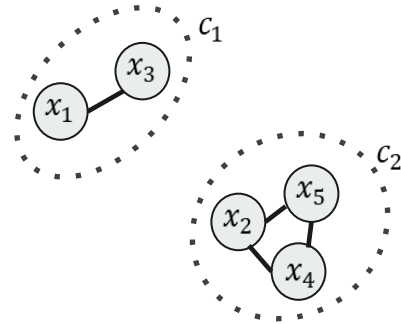
## Avoiding the use of an other clustering algorithm

• We want:  $C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$  but remember that  $1_{S_1} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$  ;  $1_{S_2} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}$

Then, if we re-write what we have:  $\phi = \begin{pmatrix} \alpha_1^{(1)} & \alpha_1^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \\ \alpha_1^{(1)} & \alpha_1^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \alpha_1^{(1)} & \alpha_1^{(2)} \\ \alpha_2^{(1)} & \alpha_2^{(2)} \end{pmatrix} := C \times A$

With  $A$  the mixing matrix. Then, if we manage to find what  $A$  is, finding  $C$  is easy:  $C = \phi A^{-1}$

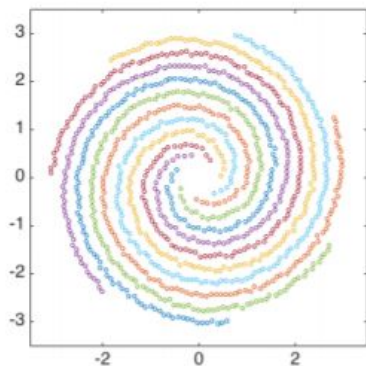
- What is  $A$  ? ....the concatenation of the linearly independent rows of  $\phi$   $\longrightarrow$  easy to find !



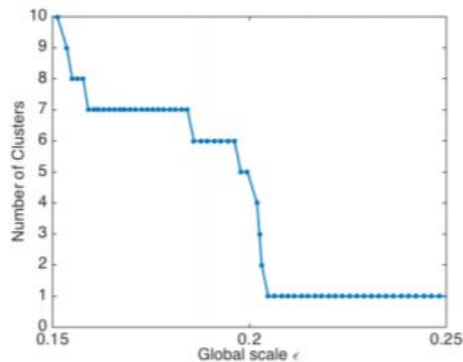
# Limits of the method and how to overcome them

Using persistence to find the right number of clusters

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



(a)



(b)

$$\epsilon \rightarrow 0 \Rightarrow k = N$$

$$\epsilon \rightarrow +\infty \Rightarrow k = 1$$



# Limits of the method and how to overcome them

## Defining a local kernel to counter the effect of non-uniform sampling

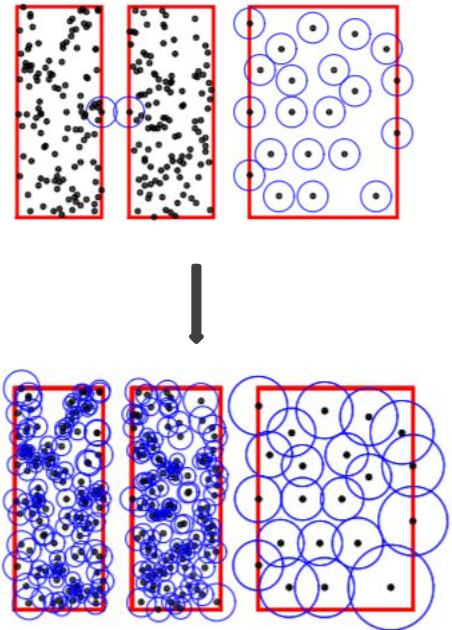
$$W_{ij} = h\left(\frac{\|x_i - x_j\|^2}{\epsilon^2}\right), \quad \text{where } h(x) = \begin{cases} 1 & \text{if } x < 1 \\ 0 & \text{otherwise.} \end{cases}$$

What we want:  $W_{ij} = 1$  if  $x_i$  similar to  $x_j$ , 0 otherwise

Solution: Define a local notion of « density »  $q$  and rescale the kernel with it !

$$W_\epsilon(x, y) = h\left(\frac{\|x - y\|^2}{\epsilon^2(q(x)q(y))^{-1/2}}\right),$$

Question: How do we find  $q$  ?



# Limits of the method and how to overcome them

## Defining a local kernel to counter the effect of non-uniform sampling

2. Find a kernel density estimate  $q(x_i)$ . For example:

(a) Define the ad hoc bandwidth function  $\hat{\rho}_i = \sqrt{\sum_{j=1}^k \|x_i - x_{I(i,j)}\|^2}$  where  $I(i, j)$  is the index of the  $j$ -th nearest neighbor of  $x_i$ .

Tune the bandwidth for the kernel density estimate in steps (b)-(f).

(b) Let  $\delta_l = 2^l$  for  $l = -30, -29.9, \dots, 9.9, 10$ .

(c) Compute  $T_l = \sum_{i,j=1}^N \exp\left(\frac{-\|x_i - x_j\|^2}{4\delta_l^2 \hat{\rho}_i \hat{\rho}_j}\right)$ .

(d) Estimate the local power law  $T_l = \delta_l^a$  at each  $l$  by  $a_l = \frac{\log T_l - \log T_{l-1}}{\log \delta_l - \log \delta_{l-1}}$ .

(e) Estimate the intrinsic dimension  $d = \max_{\delta_l} \{a_l\}$  and set  $\delta = \operatorname{argmax}_{\delta_l} \{a_l\}$ .

(f) Estimate the density

$$q_i = q(x_i) = (4\pi\delta^2 \hat{\rho}_i^2)^{-d/2} N^{-1} \sum_{j=1}^N \exp\left(\frac{-\|x_i - x_j\|^2}{4\delta^2 \hat{\rho}_i \hat{\rho}_j}\right).$$



# Conclusion

## Bibliography

- *Tyrus Berry, Timothy sauer, Spectral clustering from a geometric viewpoint* (2015)
- *Ulrike Von Luxburg, A tutorial on spectral clustering* (2007)

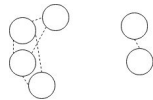


# Appendix

# Similarity graphs

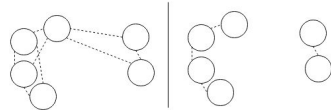
How to construct a graph matrix from a dataset of points given pairwise similarities (or distances)?

$\epsilon$ -neighborhood graph



connect if distance  $< \epsilon$

$k$ -nearest neighbor graph



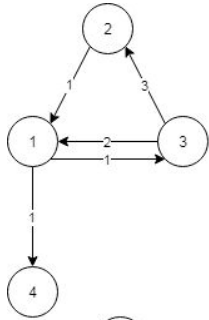
(e.g. 2-nearest, simple or mutual)

fully connected graph : connect all points with pairwise similarities  $> 0$

$k$ -nearest and fully connected  $\Rightarrow$  weight the connected edges with pairwise similarities

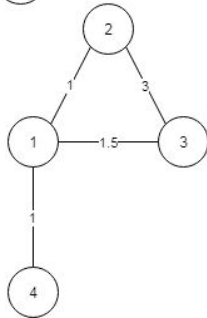
epsilon-neighborhood  $\Rightarrow$  unweighted graph

# Weighted adjacency matrices



	1	2	3	4
1			1	1
2	1			
3	2	3		
4				

$W$  Directed graph



	1	2	3	4
1		1	1.5	1
2	1		3	
3	1.5	3		
4	1			

$W$  Undirected graph

$$w_{ij} \geq 0$$

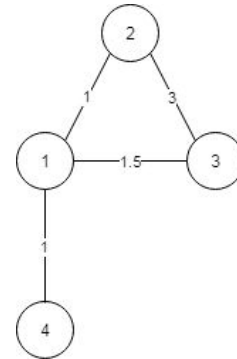
# Degree matrices

Degree of a vertex  $v_i \in V$ :  $d_i = \sum_{j=1}^n w_{ij}$

$$D = \begin{pmatrix} d_1 & & & \\ & d_2 & & \\ & & \dots & \\ & & & d_n \end{pmatrix}$$

	1	2	3	4
1				
2	1		3	
3	1.5	3		
4	1			

$d_2 = 4$



# Graph Laplacian matrices

Given an  $n \times n$  symmetric weight matrix describing the affinity between pairs of points

(for example:  $W_{ij} = 1$  if  $\|x_i - x_j\| < \epsilon$  or  $W_{ij} = s(x_i, x_j) = \exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right)$ )

$$D = W\mathbf{1}$$

$$\hat{W} = D^{-1}WD^{-1}$$

$$\hat{D} = \hat{W}\mathbf{1}$$

Unnormalized

$$L_{\text{un}} = D - W$$

Symmetric

$$L_{\text{sym}} = I - D^{-1/2}WD^{-1/2}$$

Random walk

$$L_{\text{rw}} = I - D^{-1}W$$

Diffusion map

$$L_{\text{dm}} = I - \hat{D}^{-1}\hat{W}$$

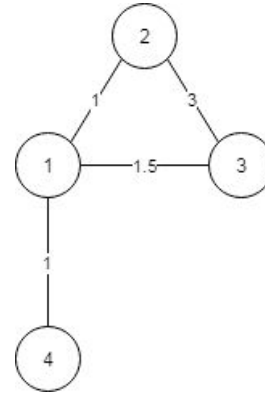


$$L = D - W$$

# Graph Laplacian matrices


$L$  has the following properties:

- $\forall \mathbf{f} \in \mathbb{R}^n, \mathbf{f}^\top \mathbf{L} \mathbf{f} = \frac{1}{2} \sum_{i,j=1}^n w_{ij} (f_i - f_j)^2$
- $L$  is symmetric and positive semi-definite
- The smallest eigenvalue is 0, the corresponding eigenvector is  $\mathbf{1}$
- $0 = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$



	1	2	3	4
1	3.5	-1	-1.5	-1
2	-1	4	-3	
3	-1.5	-3	4.5	
4	-1			1

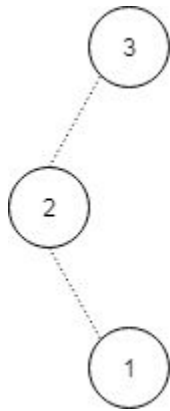
$$L = D - W$$


$$\begin{aligned} f'Lf &= f'Df - f'Wf = \sum_{i=1}^n d_i f_i^2 - \sum_{i,j=1}^n f_i f_j w_{ij} \\ &= \frac{1}{2} \left( \sum_{i=1}^n d_i f_i^2 - 2 \sum_{i,j=1}^n f_i f_j w_{ij} + \sum_{j=1}^n d_j f_j^2 \right) = \frac{1}{2} \sum_{i,j=1}^n w_{ij} (f_i - f_j)^2. \end{aligned}$$

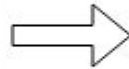
- $\Rightarrow$  L is positive definite. It is symmetric from the symmetry of W and D.
- The smallest eigenvalue is 0 with the eigenvector  $\mathbf{1}$  because  $L=D-W$  (and diagonal elements of D are sums of row elements of W)
- Since L is positive positive semi-definite, its eigenvalues are  $\geq 0$

# W-connected components

Classes of points  $x_i$  such that  $x_i \sim x_j$  if  $W_{ij}^p \neq 0$  for some  $p > 0$



0	1	0
1	0	1
0	1	0



1	0	1
0	1	0
1	0	1

$$\mathbf{1}_A = (f_1, \dots, f_n)^\top \quad \text{where} \quad f_i = \begin{cases} 1 & \text{if } i \in A \\ 0 & \text{otherwise} \end{cases}$$

$$L = D - W$$

## W-connected components

**THEOREM** - The multiplicity  $k$  of the eigenvalue  $\mathbf{0}$  of  $L$  is the number of  $W$ -connected components.

Moreover, for:  $L_{\text{un}}$ , the eigenspace associated with  $\mathbf{0}$  is spanned by the indicator functions

$$L_{\text{rw}} \quad \mathbf{1}_{S_i} \quad i = 1..k$$

$L_{\text{dm}}$  of the  $W$ -connected components  $S_i$

$$\forall \mathbf{f} \in \mathbb{R}^n, \mathbf{f}^\top \mathbf{L} \mathbf{f} = \frac{1}{2} \sum_{i,j=1}^n w_{ij} (f_i - f_j)^2$$

## W-connected components

PROOF (1/2) - with  $k=1$ , we assume that  $\mathbf{x}$  is an eigenvector of 0:

$$0 = \boldsymbol{\varphi}^\top \mathbf{L} \boldsymbol{\varphi} = \sum_{i,j=1}^n w_{ij} (\varphi_i - \varphi_j)^2 \Rightarrow (\forall i, j : w_{ij} \neq 0), \varphi_i = \varphi_j$$

=> if two vertices  $v_i, v_j$  are connected (and, recursively, if they are in the same connected component), the corresponding  $f_i$  and  $f_j$  must be equal.

$$k = 1 \Rightarrow \mathbf{f} = \mathbf{1} \quad (\text{eigenvector of } 0)$$



## W-connected components

PROOF (2/2) - with k distinct connected components

$$L \sim \begin{pmatrix} L_1 & & & \\ & L_2 & & \\ & & \ddots & \\ & & & L_k \end{pmatrix}$$

$L_i$  : graph laplacian of the i-th connected component

$$\Rightarrow \{\lambda\}_L = \bigcup_{i=1}^k \{\lambda\}_{L_i}$$

And the corresponding eigenvectors are the  $\mathbf{1}_{S_i}$