SPECTRAL AND GRAPH CLUSTERING

- clustering over graph data
- goal is to cluster the nodes by using the edges and their weights, which represent the similarity between the incident nodes.
- Graph clustering also has a very strong connection to spectral decomposition of graph-based matrices

Graphs and Matrices

Given a dataset $\boldsymbol{D} = \{\boldsymbol{x}_i\}_{i=1}^{n}$ consisting of n points in $\mathbb{R}^d$, let $\boldsymbol{A}$ denote the $n \times n$ symmetric *similarity matrix* between the points, given as

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$$

where $\boldsymbol{A}(i,j) = a_{ij}$ denotes the similarity or affinity between points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$.

We require the similarity to be symmetric and non-negative, that is, $a_{ij} = a_{ji}$ and $a_{ij} \geq 0$, respectively.

The matrix $\boldsymbol{A}$ is the *weighted adjacency matrix* for the data graph. If all affinities are 0 or 1, then $\boldsymbol{A}$ represents the regular adjacency relationship between the vertices.

example

Iris dataset

n = 150 points xi ∈ R4

$$a_{ij} = \exp\left\{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right\}$$

top q nearest neighbors

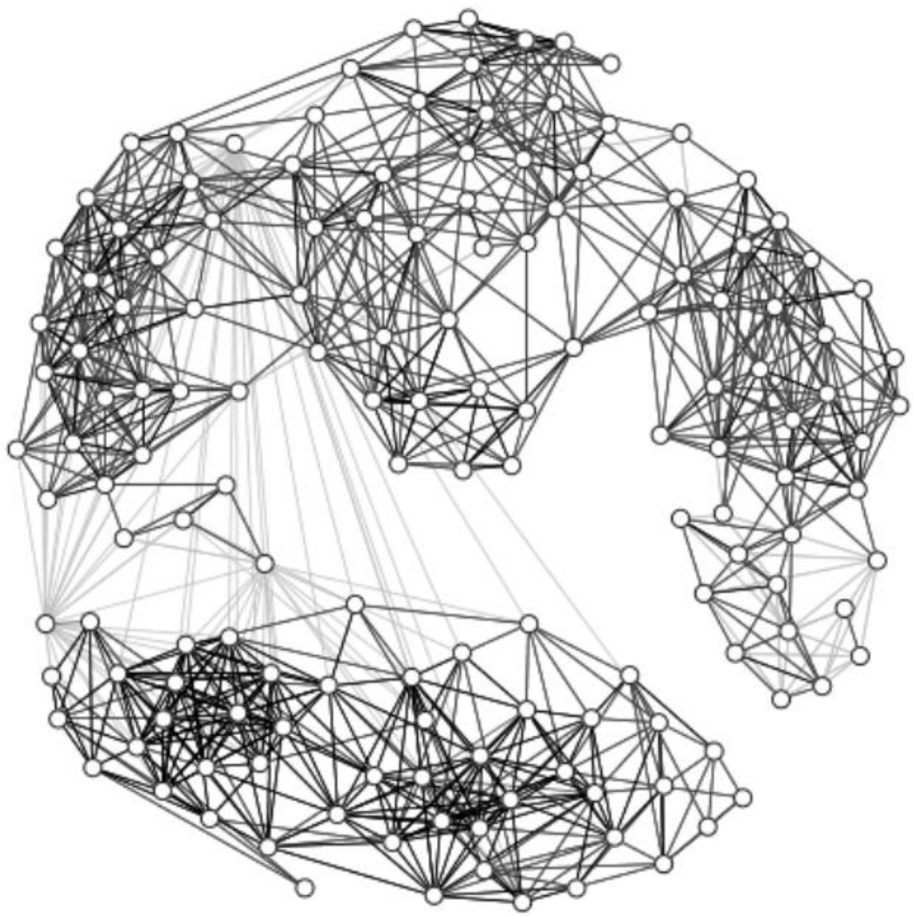
aiq represents the similarity value between xi and its qth nearest neighbor

q = 16

$$N_q(\mathbf{x}_i) = \left\{\mathbf{x}_j \in V :\ a_{ij} \leq a_{iq}\right\}$$

An edge is added between nodes xi and xj if and only if both nodes are mutual nearest neighbors

Iris Graph



Degree Matrix

For a vertex $\boldsymbol{x}_i$, let d_i denote the *degree* of the vertex, defined as

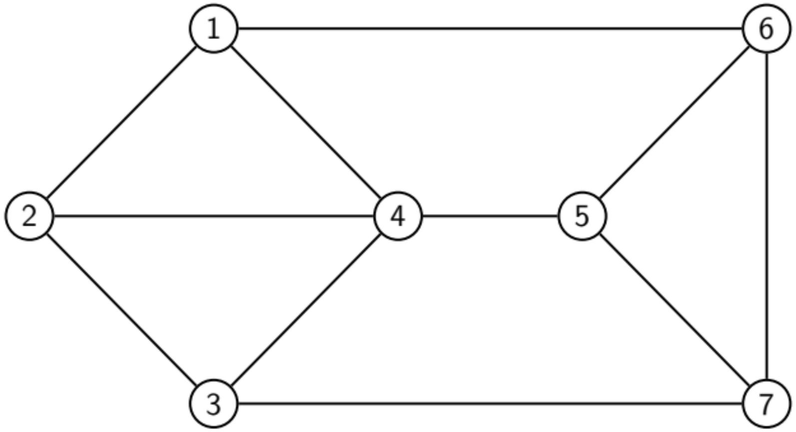
$$d_i = \sum_{j=1}^{n} a_{ij}$$

We define the *degree matrix* Δ of graph G as the $n \times n$ diagonal matrix:

$$\Delta = \begin{pmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_n \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^{n} a_{1j} & 0 & \cdots & 0 \\ 0 & \sum_{j=1}^{n} a_{2j} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sum_{j=1}^{n} a_{nj} \end{pmatrix}$$

Δ can be compactly written as $\Delta(i,i) = d_i$ for all $1 \leq i \leq n$.

example



Its adjacency and degree matrices are given as

$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \qquad \Delta = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 \end{pmatrix}$$

Graph Laplacian Matrix

The *Laplacian matrix* of a graph is defined as

$$
\begin{aligned}
\boldsymbol{L} &= \Delta - \boldsymbol{A} \\
&= \begin{pmatrix} \sum_{j=1}^{n} a_{1j} & 0 & \cdots & 0 \\ 0 & \sum_{j=1}^{n} a_{2j} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sum_{j=1}^{n} a_{nj} \end{pmatrix} - \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \\
&= \boxed{\begin{pmatrix} \sum_{j\neq 1} a_{1j} & -a_{12} & \cdots & -a_{1n} \\ -a_{21} & \sum_{j\neq 2} a_{2j} & \cdots & -a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ -a_{n1} & -a_{n2} & \cdots & \sum_{j\neq n} a_{nj} \end{pmatrix}}
\end{aligned}
$$

Graph Laplacian Matrix

- L is a symmetric, positive semidefinite matrix

$$
\begin{aligned}
\mathbf{c}^T\mathbf{L}\mathbf{c} = \mathbf{c}^T(\mathbf{\Delta} - \mathbf{A})\mathbf{c} = \mathbf{c}^T\mathbf{\Delta}\mathbf{c} - \mathbf{c}^T\mathbf{A}\mathbf{c}& \\
&= \sum_{i=1}^{n} d_i c_i^2 - \sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j a_{ij} \\
&= \frac{1}{2}\left(\sum_{i=1}^{n} d_i c_i^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j a_{ij} + \sum_{j=1}^{n} d_j c_j^2\right) \\
&= \frac{1}{2}\left(\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} c_i^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j a_{ij} + \sum_{i=j}^{n}\sum_{i=1}^{n} a_{ij} c_j^2\right) \\
&= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}(c_i - c_j)^2 \\
&\geq 0 \qquad \text{because } a_{ij} \geq 0 \text{ and } (c_i - c_j)^2 \geq 0
\end{aligned}
$$

Graph Laplacian Matrix

$$\begin{pmatrix} \sum_{j\neq 1} a_{1j} & -a_{12} & \cdots & - & a_{1n} \\ -a_{21} & \sum_{j\neq 2} a_{2j} & \cdots & - & a_{2n} \\ \vdots & \vdots & \cdots & & \vdots \\ -a_{n1} & -a_{n2} & \cdots & & \sum_{j\neq n} a_{nj} \end{pmatrix}$$

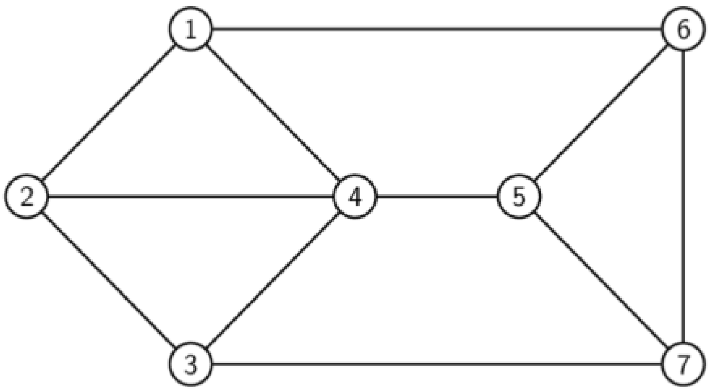
L is a symmetric, positive semidefinite matrix. This means that ***L*** has n real, non-negative eigenvalues, which can be arranged in decreasing order as follows: $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$. Because ***L*** is symmetric, its eigenvectors are orthonormal.

$$L_1 + L_2 + L_3 + \cdots + L_n = \mathbf{0}.$$

This implies that the rank of $\mathbf{L}$ is at most $n-1$, and the smallest eigenvalue is $\lambda_n = 0$

with the corresponding eigenvector given as $\mathbf{u}_n = \frac{1}{\sqrt{n}}(1, 1, \ldots, 1)^T = \frac{1}{\sqrt{n}}\mathbf{1}$

example



The graph Laplacian is given as

$$\boldsymbol{L} = \Delta - \boldsymbol{A} = \begin{pmatrix} 3 & -1 & 0 & -1 & 0 & -1 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 3 & -1 & 0 & 0 & -1 \\ -1 & -1 & -1 & 4 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & 0 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & -1 & 0 & -1 & -1 & 3 \end{pmatrix}$$

The eigenvalues of $\boldsymbol{L}$ are as follows: $\lambda_1 = 5.618$, $\lambda_2 = 4.618$, $\lambda_3 = 4.414$, $\lambda_4 = 3.382$, $\lambda_5 = 2.382$, $\lambda_6 = 1.586$, $\lambda_7 = 0$

normalized symmetric Laplacian matrix

The *normalized symmetric Laplacian matrix* of a graph is defined as

$$
\begin{aligned}
\mathbf{L}^s &= \boldsymbol{\Delta}^{-1/2}\mathbf{L}\boldsymbol{\Delta}^{-1/2} \\
&= \boldsymbol{\Delta}^{-1/2}(\boldsymbol{\Delta}-\mathbf{A})\boldsymbol{\Delta}^{-1/2} = \boldsymbol{\Delta}^{-1/2}\boldsymbol{\Delta}\boldsymbol{\Delta}^{-1/2} - \boldsymbol{\Delta}^{-1/2}\mathbf{A}\boldsymbol{\Delta}^{-1/2} \\
&= \mathbf{I} - \boldsymbol{\Delta}^{-1/2}\mathbf{A}\boldsymbol{\Delta}^{-1/2}
\end{aligned}
$$

$$
\boldsymbol{\Delta}^{1/2}(i,i) = \sqrt{d_i} \qquad \boldsymbol{\Delta}^{-1/2}(i,i) = \frac{1}{\sqrt{d_i}}
$$

$$
\boxed{
\begin{aligned}
\mathbf{L}^s &= \boldsymbol{\Delta}^{-1/2}\mathbf{L}\boldsymbol{\Delta}^{-1/2} \\
&= \begin{pmatrix}
\frac{\sum_{j\neq 1} a_{1j}}{\sqrt{d_1 d_1}} & -\frac{a_{12}}{\sqrt{d_1 d_2}} & \cdots - & \frac{a_{1n}}{\sqrt{d_1 d_n}} \\
-\frac{a_{21}}{\sqrt{d_2 d_1}} & \frac{\sum_{j\neq 2} a_{2j}}{\sqrt{d_2 d_2}} & \cdots - & \frac{a_{2n}}{\sqrt{d_2 d_n}} \\
\vdots & \vdots & \ddots & \vdots \\
-\frac{a_{n1}}{\sqrt{d_n d_1}} & -\frac{a_{n2}}{\sqrt{d_n d_2}} & \cdots & \frac{\sum_{j\neq n} a_{nj}}{\sqrt{d_n d_n}}
\end{pmatrix}
\end{aligned}
}
$$

$$\mathbf{c}^T\mathbf{L}^s\mathbf{c} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}\left(\frac{c_i}{\sqrt{d_i}} - \frac{c_j}{\sqrt{d_j}}\right)^2 \geq 0$$

$\mathbf{L}^s$ is also positive semidefinite

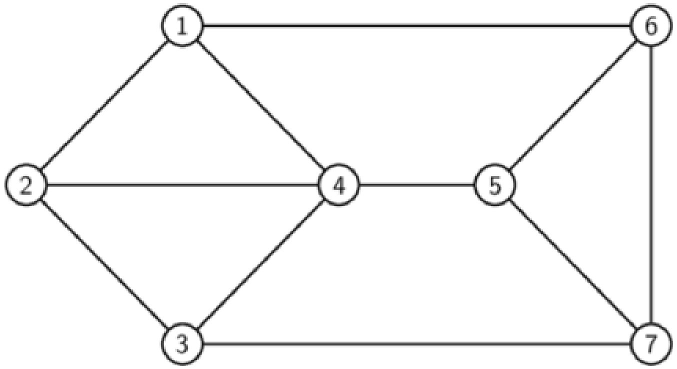
if L_i^s denotes the ith column of $\mathbf{L}^s$

$$\sqrt{d_1}L_1^s + \sqrt{d_2}L_2^s + \sqrt{d_3}L_3^s + \cdots + \sqrt{d_n}L_n^s = \mathbf{0}$$

$\mathbf{L}^s$ has rank at most $n-1$, with the smallest eigenvalue $\lambda_n = 0$,

$$\frac{1}{\sqrt{\sum_i d_i}}(\sqrt{d_1}, \sqrt{d_2}, \ldots, \sqrt{d_n})^T = \frac{1}{\sqrt{\sum_i d_i}}\mathbf{\Delta}^{1/2}\mathbf{1}$$

example



The normalized symmetric Laplacian is given as

$$\boldsymbol{L}^s = \begin{pmatrix} 1 & -0.33 & 0 & -0.29 & 0 & -0.33 & 0 \\ -0.33 & 1 & -0.33 & -0.29 & 0 & 0 & 0 \\ 0 & -0.33 & 1 & -0.29 & 0 & 0 & -0.33 \\ -0.29 & -0.29 & -0.29 & 1 & -0.29 & 0 & 0 \\ 0 & 0 & 0 & -0.29 & 1 & -0.33 & -0.33 \\ -0.33 & 0 & 0 & 0 & -0.33 & 1 & -0.33 \\ 0 & 0 & -0.33 & 0 & -0.33 & -0.33 & 1 \end{pmatrix}$$

The eigenvalues of $\boldsymbol{L}^s$ are as follows: $\lambda_1 = 1.7$, $\lambda_2 = 1.539$, $\lambda_3 = 1.405$, $\lambda_4 = 1.045$, $\lambda_5 = 0.794$, $\lambda_6 = 0.517$, $\lambda_7 = 0$

CLUSTERING AS GRAPH CUTS

A *k-way cut* in a graph is a partitioning or clustering of the vertex set, given as $\mathcal{C} = \{C_1, \ldots, C_k\}$. We require $\mathcal{C}$ to optimize some objective function that captures the intuition that nodes within a cluster should have high similarity, and nodes from different clusters should have low similarity.

Given $S \subseteq V$, we denote by $\overline{S}$ the complementary set of vertices, that is, $\overline{S} = V - S$. A *(vertex) cut* in a graph is defined as a partitioning of V into $S \subset V$ and $\overline{S}$.

CLUSTERING AS GRAPH CUTS

Given a weighted graph G defined by its similarity matrix $\boldsymbol{A}$, let $S, T \subseteq V$ be any two subsets of the vertices. We denote by $W(S, T)$ the sum of the weights on all edges with one vertex in S and the other in T, given as

$$\boxed{W(S,T) = \sum_{v_i \in S} \sum_{v_j \in T} a_{ij}}$$

The *weight of the cut* or *cut weight* is defined as the sum of all the weights on edges between vertices in S and $\overline{S}$, given as $W(S, \overline{S})$.

CLUSTERING AS GRAPH CUTS

Given a clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$ comprising k clusters. Let $\mathbf{c}_i \in \{0,1\}^n$ be the *cluster indicator vector* that records the cluster membership for cluster C_i, defined as

$$c_{ij} = \begin{cases} 1 & \text{if } v_j \in C_i \\ 0 & \text{if } v_j \notin C_i \end{cases}$$

The cluster size can be written as

$$\boxed{|C_i| = \mathbf{c}_i^T \mathbf{c}_i = \|\mathbf{c}_i\|^2}$$

$$\mathbf{c}_i^T \mathbf{c}_j = 0$$

The *volume* of a cluster C_i is defined as the sum of all the weights on edges with one end in cluster C_i:

$$\boxed{vol(C_i) = \sum_{v_j \in C_i} d_j = \sum_{v_j \in C_i} \sum_{v_r \in V} a_{jr} = W(C_i, V)}$$

CLUSTERING AS GRAPH CUTS

$$vol(C_i) = W(C_i, V) \qquad = \sum_{v_r \in C_i} d_r = \sum_{v_r \in C_i} c_{ir} d_r c_{ir} = \sum_{r=1}^{n} \sum_{s=1}^{n} c_{ir} \mathbf{\Delta}_{rs} c_{is}$$

$$\boxed{vol(C_i) = \mathbf{c}_i^T \mathbf{\Delta} \mathbf{c}_i}$$

Consider the sum of weights of all internal edges:

$$W(C_i, C_i) = \sum_{v_r \in C_i} \sum_{v_s \in C_i} a_{rs} = \sum_{r=1}^{n} \sum_{s=1}^{n} c_{ir} a_{rs} c_{is}$$

We can therefore rewrite the sum of internal weights as

$$\boxed{W(C_i, C_i) = \mathbf{c}_i^T \mathbf{A} \mathbf{c}_i}$$

CLUSTERING AS GRAPH CUTS

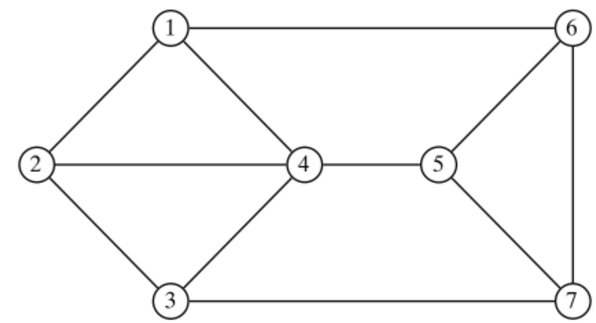
$$W(C_i, \overline{C_i}) = \sum_{v_r \in C_i} \sum_{v_s \in V - C_i} a_{rs} = W(C_i, V) - W(C_i, C_i)$$

$$= \mathbf{c}_i(\mathbf{\Delta} - \mathbf{A})\mathbf{c}_i = \mathbf{c}_i^T \mathbf{L} \mathbf{c}_i$$

Example

$C_1 = \{1, 2, 3, 4\}$ $C_2 = \{5, 6, 7\}$

$\mathbf{c}_1 = (1, 1, 1, 1, 0, 0, 0)^T$ $\mathbf{c}_2 = (0, 0, 0, 0, 1, 1, 1)^T$



$W(C_1, \overline{C_1}) = W(C_1, C_2) = 3$

$$
\begin{aligned}
W(C_1, \overline{C_1}) &= \mathbf{c}_1^T \mathbf{L} \mathbf{c}_1 \\
&= (1, 1, 1, 1, 0, 0, 0) \begin{pmatrix} 3 & -1 & 0 & -1 & 0 & -1 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 3 & -1 & 0 & 0 & -1 \\ -1 & -1 & -1 & 4 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & 0 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & -1 & 0 & -1 & -1 & 3 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \\
&= (1, 0, 1, 1, -1, -1, -1)(1, 1, 1, 1, 0, 0, 0)^T = 3
\end{aligned}
$$

Clustering Objective Functions: Ratio Cut

The clustering objective function can be formulated as an optimization problem over the k-way cut $\mathcal{C} = \{C_1, \ldots, C_k\}$.

Ratio Cut

The *ratio cut* objective is defined over a k-way cut as follows:

$$\boxed{\min_{\mathcal{C}} J_{rc}(\mathcal{C}) = \sum_{i=1}^{k} \frac{W(C_i, \overline{C_i})}{|C_i|} = \sum_{i=1}^{k} \frac{\mathbf{c}_i^T \mathbf{L} \mathbf{c}_i}{\mathbf{c}_i^T \mathbf{c}_i} = \sum_{i=1}^{k} \frac{\mathbf{c}_i^T \mathbf{L} \mathbf{c}_i}{\|\mathbf{c}_i\|^2}}$$

for binary cluster indicator vectors ci, the ratio cut objective is NP-hard

Clustering Objective Functions: Ratio Cut

An obvious relaxation is to allow c_i to take on any real value. In this case, we can rewrite the objective as

$$\min_{\mathcal{C}} J_{rc}(\mathcal{C}) = \sum_{i=1}^{k} \frac{\mathbf{c}_i^T \mathbf{L} \mathbf{c}_i}{\|\mathbf{c}_i\|^2} = \sum_{i=1}^{k} \left(\frac{\mathbf{c}_i}{\|\mathbf{c}_i\|}\right)^T \mathbf{L} \left(\frac{\mathbf{c}_i}{\|\mathbf{c}_i\|}\right) = \sum_{i=1}^{k} \mathbf{u}_i^T \mathbf{L} \mathbf{u}_i$$

$$\mathbf{u}_i^T \mathbf{u}_i = 1$$

Clustering Objective Functions: Ratio Cut

we introduce the Lagrange multiplier λ_i

for each cluster C_i. We have

$$\frac{\partial}{\partial \mathbf{u}_i}\left(\sum_{i=1}^{k} \mathbf{u}_i^T \mathbf{L} \mathbf{u}_i + \sum_{i=1}^{n} \lambda_i (1 - \mathbf{u}_i^T \mathbf{u}_i)\right) = \mathbf{0}, \text{ which implies that}$$

$$2\mathbf{L}\mathbf{u}_i - 2\lambda_i \mathbf{u}_i = \mathbf{0}, \text{ and thus}$$

$$\mathbf{L}\mathbf{u}_i = \lambda_i \mathbf{u}_i$$

Eigen vector

$$\mathbf{u}_i^T \mathbf{L} \mathbf{u}_i = \mathbf{u}_i^T \lambda_i \mathbf{u}_i = \lambda_i$$

Clustering Objective Functions: Ratio Cut

we should choose the k smallest eigenvalues, and the corresponding eigenvectors, so that

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$$

$$\min_{\mathcal{C}} J_{rc}(\mathcal{C}) = \mathbf{u}_n^T \mathbf{L} \mathbf{u}_n + \cdots + \mathbf{u}_{n-k+1}^T \mathbf{L} \mathbf{u}_{n-k+1}$$

$$= \lambda_n + \cdots + \lambda_{n-k+1}$$

$$0 = \lambda_n \leq \lambda_{n-1} \leq \lambda_{n-k+1}$$

Normalized cut

Normalized cut is similar to ratio cut, except that it divides the cut weight of each cluster by the volume of a cluster instead of its size

$$\boxed{\min_{\mathcal{C}} J_{nc}(\mathcal{C}) = \sum_{i=1}^{k} \frac{W(C_i, \overline{C_i})}{vol(C_i)} = \sum_{i=1}^{k} \frac{\mathbf{c}_i^T \mathbf{L} \mathbf{c}_i}{\mathbf{c}_i^T \boldsymbol{\Delta} \mathbf{c}_i}}$$

$$\begin{aligned}
\min_{\mathcal{C}} J_{nc}(\mathcal{C}) &= \sum_{i=1}^{k} \frac{\mathbf{c}_i^T \mathbf{L} \mathbf{c}_i}{\mathbf{c}_i^T \boldsymbol{\Delta} \mathbf{c}_i} = \sum_{i=1}^{k} \frac{\mathbf{c}_i^T \left(\boldsymbol{\Delta}^{1/2} \boldsymbol{\Delta}^{-1/2}\right) \mathbf{L} \left(\boldsymbol{\Delta}^{-1/2} \boldsymbol{\Delta}^{1/2}\right) \mathbf{c}_i}{\mathbf{c}_i^T \left(\boldsymbol{\Delta}^{1/2} \boldsymbol{\Delta}^{1/2}\right) \mathbf{c}_i} \\
&= \sum_{i=1}^{k} \frac{(\boldsymbol{\Delta}^{1/2} \mathbf{c}_i)^T (\boldsymbol{\Delta}^{-1/2} \mathbf{L} \boldsymbol{\Delta}^{-1/2})(\boldsymbol{\Delta}^{1/2} \mathbf{c}_i)}{(\boldsymbol{\Delta}^{1/2} \mathbf{c}_i)^T (\boldsymbol{\Delta}^{1/2} \mathbf{c}_i)} \\
&= \sum_{i=1}^{k} \left(\frac{\boldsymbol{\Delta}^{1/2} \mathbf{c}_i}{\|\boldsymbol{\Delta}^{1/2} \mathbf{c}_i\|} \right)^T \mathbf{L}^s \left(\frac{\boldsymbol{\Delta}^{1/2} \mathbf{c}_i}{\|\boldsymbol{\Delta}^{1/2} \mathbf{c}_i\|} \right) = \sum_{i=1}^{k} \mathbf{u}_i^T \mathbf{L}^s \mathbf{u}_i
\end{aligned}$$

Spectral Clustering Algorithm

problem we face is that the eigenvectors u_i are not binary, and thus it is not immediately clear how we can assign points to clusters

$\mathbf{u}_n, \mathbf{u}_{n-1}, \ldots, \mathbf{u}_{n-k+1}$ relaxed cluster indicator vector

One solution to this problem is to treat the $n \times k$ matrix of eigenvectors as a new data matrix:

$$\boldsymbol{U} = \begin{pmatrix} | & | & & | \\ \boldsymbol{u}_n & \boldsymbol{u}_{n-1} & \cdots & \boldsymbol{u}_{n-k+1} \\ | & | & & | \end{pmatrix} \rightarrow \text{normalize rows} \rightarrow \begin{pmatrix} — & \boldsymbol{y}_1^T & — \\ — & \boldsymbol{y}_2^T & — \\ & \vdots & \\ — & \boldsymbol{y}_n^T & — \end{pmatrix} = \boldsymbol{Y}$$

We then cluster the new points in $\boldsymbol{Y}$ into k clusters via the K-means algorithm or any other fast clustering method to obtain binary cluster indicator vectors $\boldsymbol{c}_i$.

Spectral Clustering $(\boldsymbol{D}, k)$**:**

Compute the similarity matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$

if *ratio cut* **then** $\boldsymbol{B} \leftarrow \boldsymbol{L}$

else if *normalized cut* **then** $\boldsymbol{B} \leftarrow \boldsymbol{L}^s$ or $\boldsymbol{L}^a$

Solve $\boldsymbol{B}\boldsymbol{u}_i = \lambda_i \boldsymbol{u}_i$ for $i = n, \ldots, n-k+1$, where
$\lambda_n \leq \lambda_{n-1} \leq \cdots \leq \lambda_{n-k+1}$

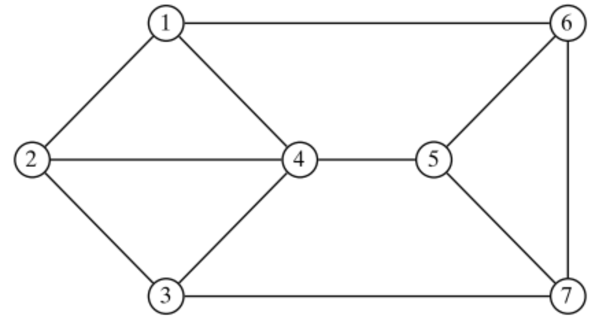
$\boldsymbol{U} \leftarrow \begin{pmatrix} \boldsymbol{u}_n & \boldsymbol{u}_{n-1} & \cdots & \boldsymbol{u}_{n-k+1} \end{pmatrix}$

$\boldsymbol{Y} \leftarrow$ normalize rows of $\boldsymbol{U}$

$\mathcal{C} \leftarrow \{C_1, \ldots, C_k\}$ via K-means on $\boldsymbol{Y}$

example

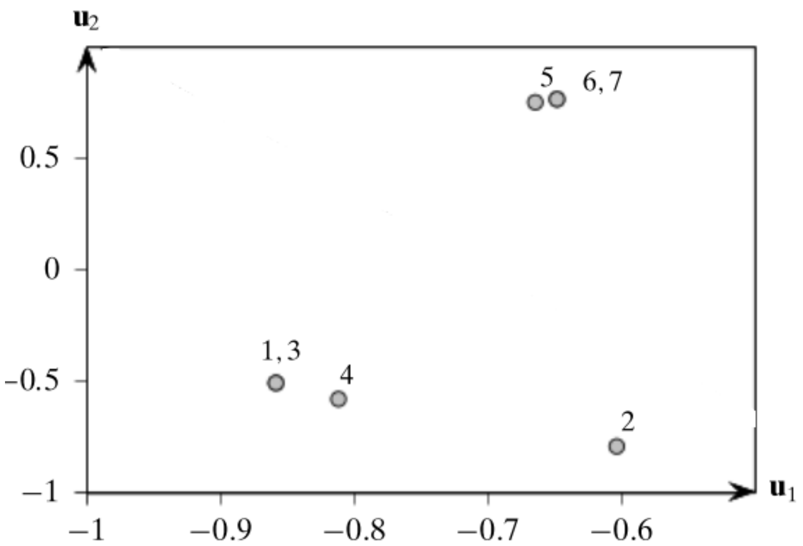
we want to find k = 2 clusters. For the normalized asymmetric Laplacian matrix , we compute the eigenvectors, v_7
and v_6, corresponding to the two smallest eigenvalues, $\lambda_7 = 0$ and $\lambda_6 = 0.517$.



$$
\mathbf{U} = \begin{array}{c} \begin{array}{cc} \mathbf{u}_1 & \mathbf{u}_2 \end{array} \\ \begin{pmatrix} -0.378 & -0.226 \\ -0.378 & -0.499 \\ -0.378 & -0.226 \\ -0.378 & -0.272 \\ -0.378 & 0.425 \\ -0.378 & 0.444 \\ -0.378 & 0.444 \end{pmatrix} \end{array}
$$

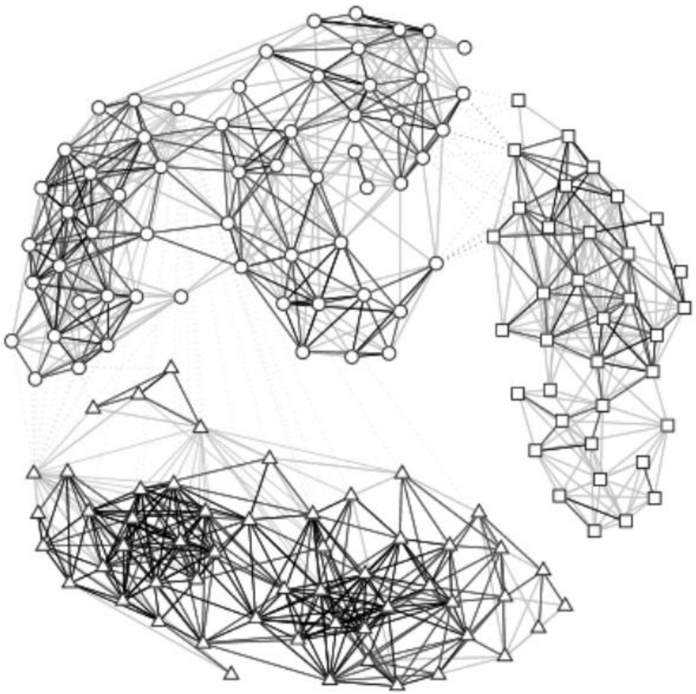
example

$$\mathbf{Y} = \begin{pmatrix} -0.859 & -0.513 \\ -0.604 & -0.797 \\ -0.859 & -0.513 \\ -0.812 & -0.584 \\ -0.664 & 0.747 \\ -0.648 & 0.761 \\ -0.648 & 0.761 \end{pmatrix}$$



example

- apply spectral clustering on the Iris graph: k = 3 clusters
- Normalized Cut
- Comparing them with the true Iris classes



	setosa	virginica	versicolor
C_1 (triangle)	50	0	4
C_2 (square)	0	36	0
C_3 (circle)	0	14	46