Spectral clustering

吕俊成

主要参考文献：A Tutorial on Spectral Clustering

Agenda

- Brief Clustering Review
- Similarity Graph
- Graph Laplacian
- Spectral Clustering Algorithm
- Graph Cut Point of View
- Practical Details

- Brief Clustering Review
- Similarity Graph
- Graph Laplacian
- Spectral Clustering Algorithm
- Graph Cut Point of View
- Practical Details

Clustering

Groups together "similar" instances in the data sample

Basic clustering problem:

- distribute data into *k* different groups such that data points similar to each other are in the same group
- Similarity between data points is defined in terms of some distance metric (can be chosen)

Clustering is useful for:

- **Similarity/Dissimilarity analysis**
  Analyze what data points in the sample are close to each other
- **Dimensionality reduction**
  High dimensional data replaced with a group (cluster) label

K-MEANS CLUSTERING

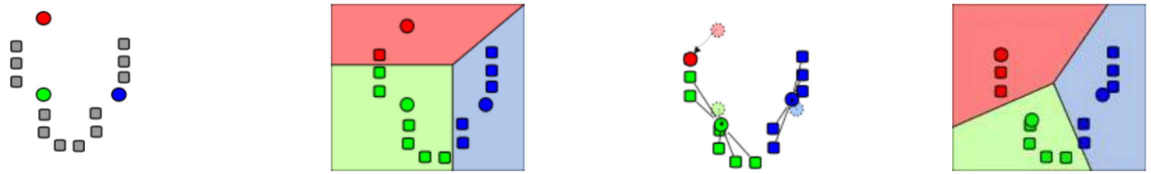
- **Description**

Given a set of observations $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$, where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into k sets $(k \leq n)$ $\mathbf{S} = \{S_1, S_2, \ldots, S_k\}$ so as to minimize the within-cluster sum of squares (WCSS):

$$\underset{S}{\arg\min} \sum_{i=1}^{k} \sum_{x_j \in S_i} \left\| x_j - u_i \right\|^2$$

where $\boldsymbol{\mu}_i$ is the mean of points in S_i.

- **Standard Algorithm**



1) k initial "means" (in this case k=3) are randomly selected from the data set.

2) k clusters are created by associating every observation with the nearest mean.

3) The centroid of each of the k clusters becomes the new means.

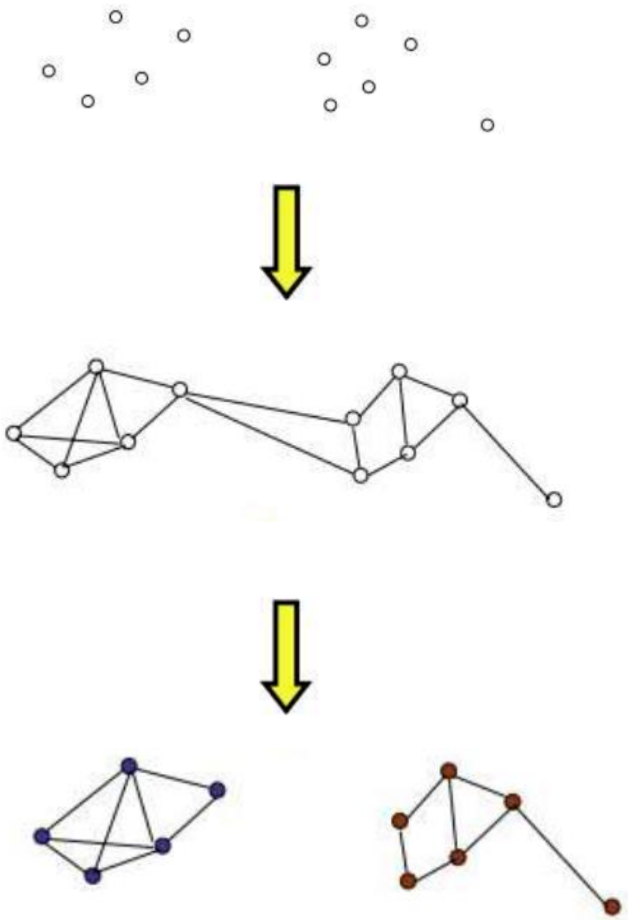
4) Steps 2 and 3 are repeated until convergence has been reached.

- Brief Clustering Review
- Similarity Graph
- Graph Laplacian
- Spectral Clustering Algorithm
- Graph Cut Point of View
- Practical Details

GENERAL

First - graph representation of data
(largely, application dependent)

Then - graph partitioning

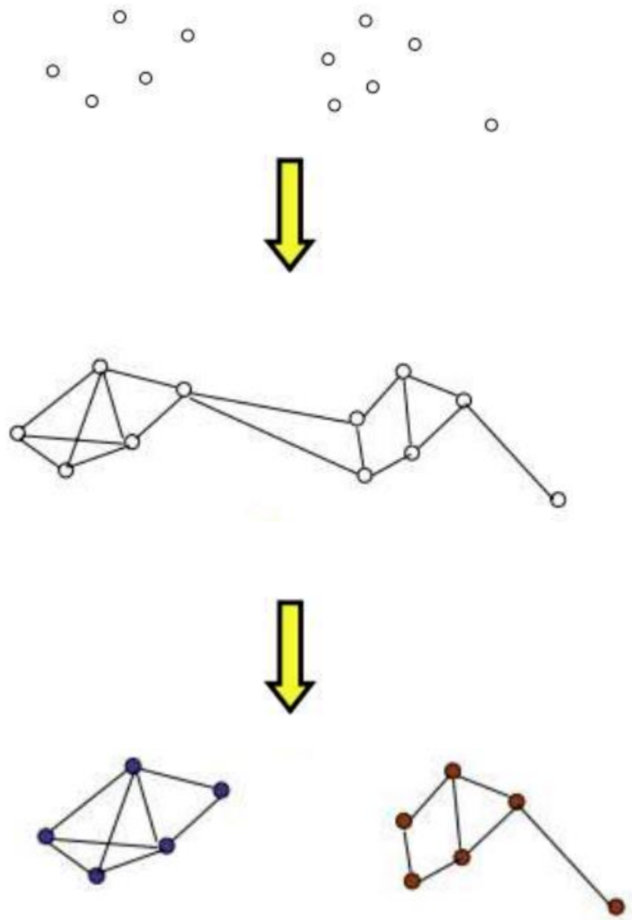


Disconnected graph components ⟷ Groups of points(Weakly connections in between components Strongly connections within components)

GENERAL

First - graph representation of data
(largely, application dependent)

Then - graph partitioning



Disconnected graph components ⟷ Groups of points(Weakly connections in between components
Strongly connections within components)

Similarity Graph

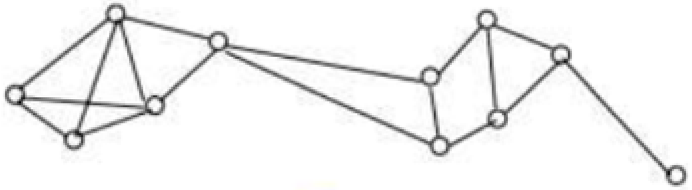
set of n data points $x_1, \ldots, x_n$

with similarites $s_{i,j} \geq 0$ between all pairs of data points $x_i, \ldots, x_j$

$\Updownarrow$ represent in a similarity graph $G = (V, E)$

each vertex v_i represents a data point x_i

two vertices are connected, if $s_{i,j}$ is larger than a certain threshold, and the edge ist weighted by $s_{i,j}$

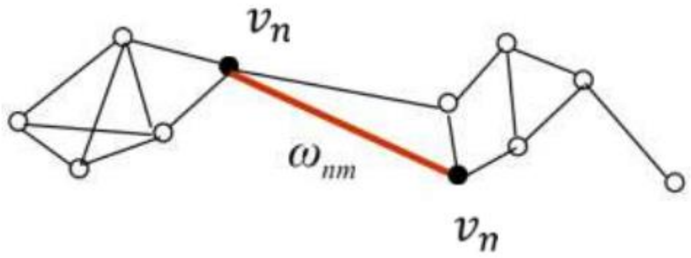


we want to find a partition of the graph such that **the edges between different groups have very low weights** (which means that points in different clusters are dissimilar from each other) **and the edges within a group have high weights** (which means that points within the same cluster are similar to each other).

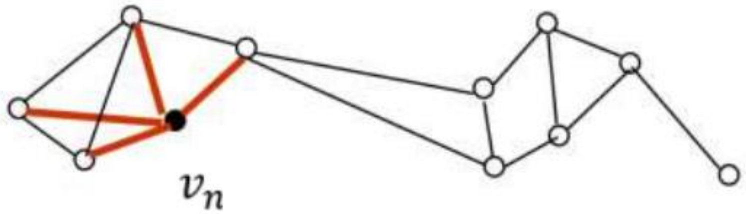
Graph notation

$G=(V,E)$:

- Vertex set $\quad V = \{v_1, \dots, v_n\}$
- Weighted adjacency matrix $\quad W = (w_{ij})\ i,j = 1, \dots, n \quad w_{ij} \geq 0$



- Degree $\quad d_i = \sum_{j=1}^{n} w_{ij}$



$$D := \begin{pmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{pmatrix}.$$

- Degree matrix $\quad$ Diagonal matrix with the degrees $d_1, \dots, d_n$ on the diagonal.

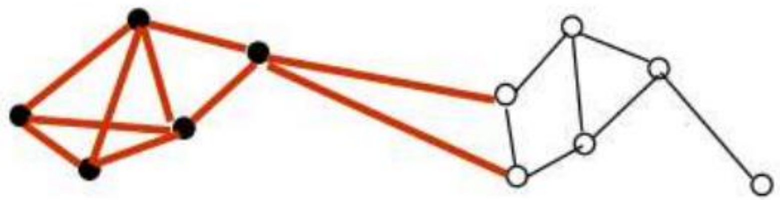
Graph notation

$G=(V,E)$:

- Indicator Vector $\mathbb{1}_A = (f_1, \dots, f_n)' \in \mathbb{R}^n$ $f_i \in \{0,1\}$
- "Size" of a subset $A \subset V$

$|A| :=$ the number of vertices in A

$vol(A) := \sum_{i \in A} d_i$



- Connected A subset A of a graph is connected if any two vertices in A can be joined by a path such that all intermediate points also lie in A.
- Connected Component it is connected and if there are no connections between vertices in A and $\bar{A}$. The nonempty sets $A_1, \dots, A_k$ form a partition of the graph if $A_i \cap A_j = \emptyset$ and $A_1 \cup \dots \cup A_k = V$.

SIMILARITY GRAPH

- ε-neighborhood graph

Connect all points whose pairwise distances are smaller than ε

$$W_{ij} = \begin{cases} 0 & s_{ij} > \epsilon \\ \epsilon & s_{ij} \leq \epsilon \end{cases}$$

- k-nearest neighbor graph

Connect vertex v_i with vertex v_j if v_j is among the k-nearest neighbors of v_i.

$$W_{ij} = W_{ji} = \begin{cases} 0 & x_i \notin KNN(x_j) \ and \ x_j \notin KNN(x_i) \\ exp(-\frac{||x_i - x_j||_2^2}{2\sigma^2}) & x_i \in KNN(x_j) \ or \ x_j \in KNN(x_i) \end{cases}$$

- fully connected graph

Connect all points with positive similarity with each other

$$W_{ij} = S_{ij} = exp(-\frac{||x_i - x_j||_2^2}{2\sigma^2})$$

All the above graphs are regularly used in spectral clustering!

- Brief Clustering Review
- Similarity Graph
- Graph Laplacian
- Spectral Clustering Algorithm
- Graph Cut Point of View
- Practical Details

- **Unnormalized Graph Laplacian**

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

$$L = D - W$$

For every vector $f \in \mathbb{R}^n$ *we have*

$$f'Lf = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2.$$

By the definition of d_i,

$$f'Lf = f'Df - f'Wf = \sum_{i=1}^{n} d_i f_i^2 - \sum_{i,j=1}^{n} f_i f_j w_{ij}$$

$$= \frac{1}{2}\left(\sum_{i=1}^{n} d_i f_i^2 - 2\sum_{i,j=1}^{n} f_i f_j w_{ij} + \sum_{j=1}^{n} d_j f_j^2\right) = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2.$$

GRAPH LAPLACIANS

- **Unnormalized Graph Laplacian**

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

$$L = D - W$$

Proposition 1 (Properties of L) *The matrix L satisfies the following properties:*

1. *For every vector $f \in \mathbb{R}^n$ we have*

$$f'Lf = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2.$$

2. *L is symmetric and positive semi-definite.*
3. *The smallest eigenvalue of L is 0, the corresponding eigenvector is the constant one vector $\mathbb{1}$.*
4. *L has n non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$.*

- **Unnormalized Graph Laplacian**

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

$$L = D - W$$

Proposition 2 (Number of connected components and the spectrum of L) *Let G be an undirected graph with non-negative weights. Then the multiplicity k of the eigenvalue 0 of L equals the number of connected components $A_1, \ldots, A_k$ in the graph. The eigenspace of eigenvalue 0 is spanned by the indicator vectors $\mathbb{1}_{A_1}, \ldots, \mathbb{1}_{A_k}$ of those components.*

Proof:

When $k = 1$, a graph consisting of only one connected component we thus only have the constant one vector $\mathbb{1}$ as eigenvector with eigenvalue 0, which obviously is the indicator vector of the connected component.

When $k > 1$, L can be written in a block form. the spectrum of L is given by the union of the spectra of L_i, and the corresponding eigenvectors of L are the eigenvectors of L_i, filled with 0 at the positions of the other blocks.

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

- Normalized Graph Laplacian

$$L_{sym} := D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$$

$$L_{rw} := D^{-1} L = I - D^{-1} W$$

We denote the first matrix by L_{sym} as it is a symmetric matrix, and the second one by L_{rw} as it is closely related to a random walk.

- Normalized Graph Laplacian

$$L_{sym} := D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$$
$$L_{rw} := D^{-1} L = I - D^{-1} W$$

Proposition 3 (Properties of L_{sym} and L_{rw}) The normalized Laplacians statisfy the following properties:

1. For every $f \in \mathbb{R}^n$ we have $$f' L_{sym} f = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}^2 \left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2$$
2. *λ is an eigenvalue of L_{rw} with eigenvector u if and only if λ is an eigenvalue of L_{sym} with eigenvector $w = D^{1/2} u$.*
3. λ is an eigenvalue of L_{rw} with eigenvector u if and only if λ and u solve the generalized eigen problem $Lu = \lambda Du$.
4. 0 is an eigenvalue of L_{rw} with the constant one vector $\mathbb{1}$ as eigenvector. 0 is an eigenvalue of L_{sym} with eigenvector $D^{1/2}\mathbb{1}$.
5. L_{sym} and L_{rw} are positive semi-definite and have n non-negative real-valued eigenvalues $0 = \lambda_i \leq \lambda_2 \leq \cdots \leq \lambda_n$.

- Normalized Graph Laplacian

$$L_{sym} := D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$$

$$L_{rw} := D^{-1} L = I - D^{-1} W$$

Proposition 4 (Number of connected components and spectra of L_{sym} and L_{rw}) Let G be an undirected graph with non-negative weights. Then the multiplicity k of the eigenvalue 0 of both L_{sym} and L_{rw} equals the number of connected components $A_1, \dots, A_k$ in the graph. For L_{rw} the eigenspace of 0 is spanned by the indicator vectors $\mathbb{1}_{A_i}$ of those components. For L_{sym}, the eigenspace of 0 is spanned by the vectors $D^{1/2} \mathbb{1}_{A_i}$.

- Brief Clustering Review
- Similarity Graph
- Graph Laplacian
- Spectral Clustering Algorithm
- Graph Cut Point of View
- Practical Details

- Unnormalized Graph Laplacian

$$L = D - W$$

Unnormalized spectral clustering

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.

- Construct a similarity graph by one of the ways described in Section 2. Let W be its weighted adjacency matrix.
- Compute the unnormalized Laplacian L.
- **Compute the first k eigenvectors $u_1, \ldots, u_k$ of L.**
- Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns.
- For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of U.
- Cluster the points $(y_i)_{i=1,\ldots,n}$ in $\mathbb{R}^k$ with the k-means algorithm into clusters $C_1, \ldots, C_k$.

Output: Clusters $A_1, \ldots, A_k$ with $A_i = \{j \mid y_j \in C_i\}$.

电子科技大学

SMILE LAB

- Normalized Graph Laplacian

$$L_{rw} := D^{-1}L = I - D^{-1}W$$

Normalized spectral clustering according to Shi and Malik (2000)

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.

- Construct a similarity graph by one of the ways described in Section 2. Let W be its weighted adjacency matrix.
- Compute the unnormalized Laplacian L.
- **Compute the first k generalized eigenvectors $u_1, \ldots, u_k$ of the generalized eigenproblem $Lu = \lambda Du$.**
- Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns.
- For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of U.
- Cluster the points $(y_i)_{i=1,\ldots,n}$ in $\mathbb{R}^k$ with the k-means algorithm into clusters $C_1, \ldots, C_k$.

Output: Clusters $A_1, \ldots, A_k$ with $A_i = \{j |\ y_j \in C_i\}$.

- Normalized Graph Laplacian

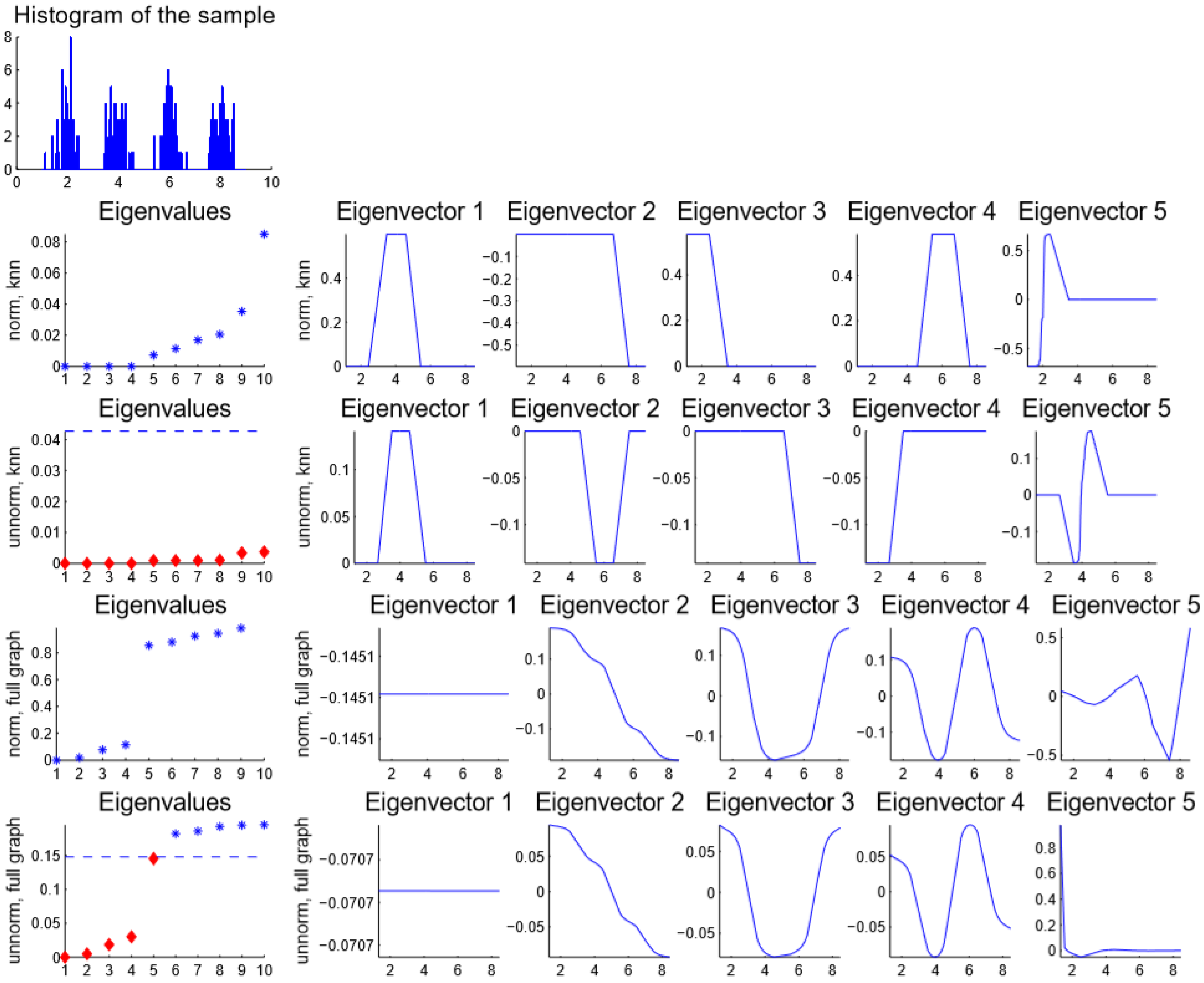
$$L_{sym} := D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$$

Normalized spectral clustering according to Ng, Jordan, and Weiss (2002)

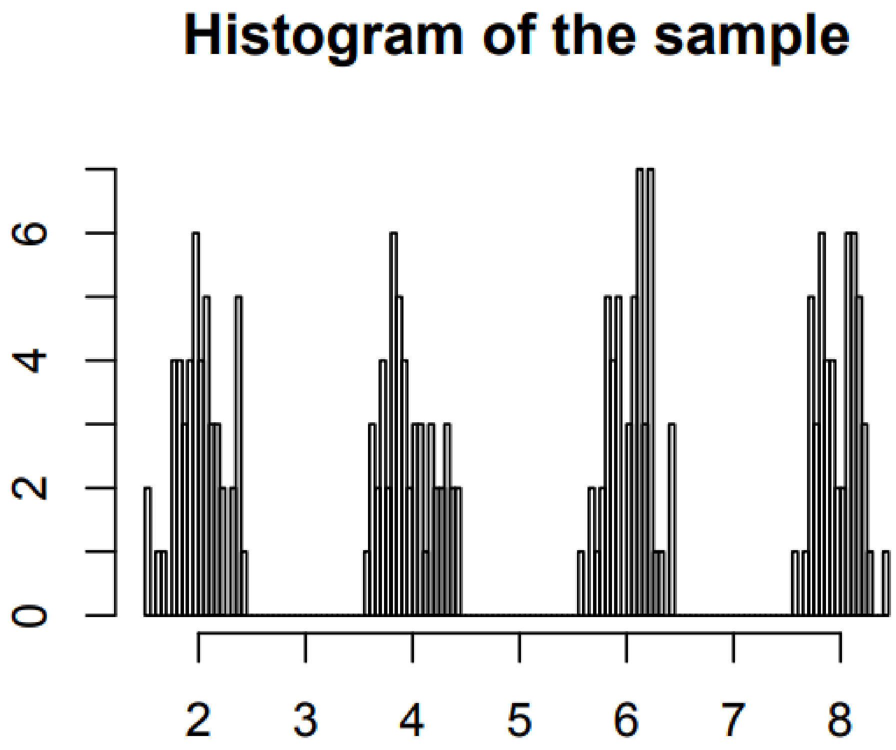
Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.

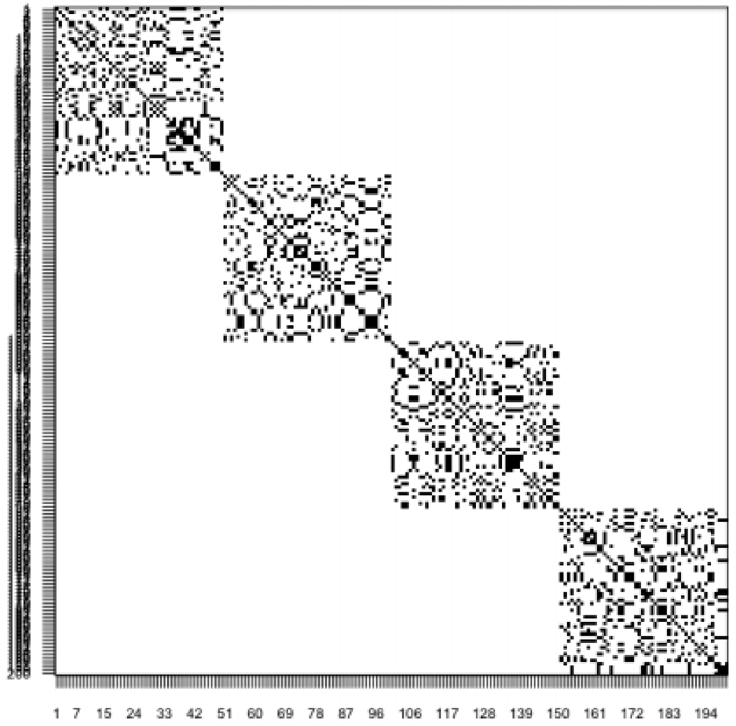
- Construct a similarity graph by one of the ways described in Section 2. Let W be its weighted adjacency matrix.
- Compute the normalized Laplacian L_{sym}.
- **Compute the first k eigenvectors $u_1, \ldots, u_k$ of L_{sym}.**
- Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns.
- **Form the matrix $T \in \mathbb{R}^{n \times k}$ from U by normalizing the rows to norm 1,** that is set $t_{ij} = u_{ij} / (\sum_k u_{ik}^2)^{1/2}$.
- For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of T.
- Cluster the points $(y_i)_{i=1,\ldots,n}$ with the k-means algorithm into clusters $C_1, \ldots, C_k$.

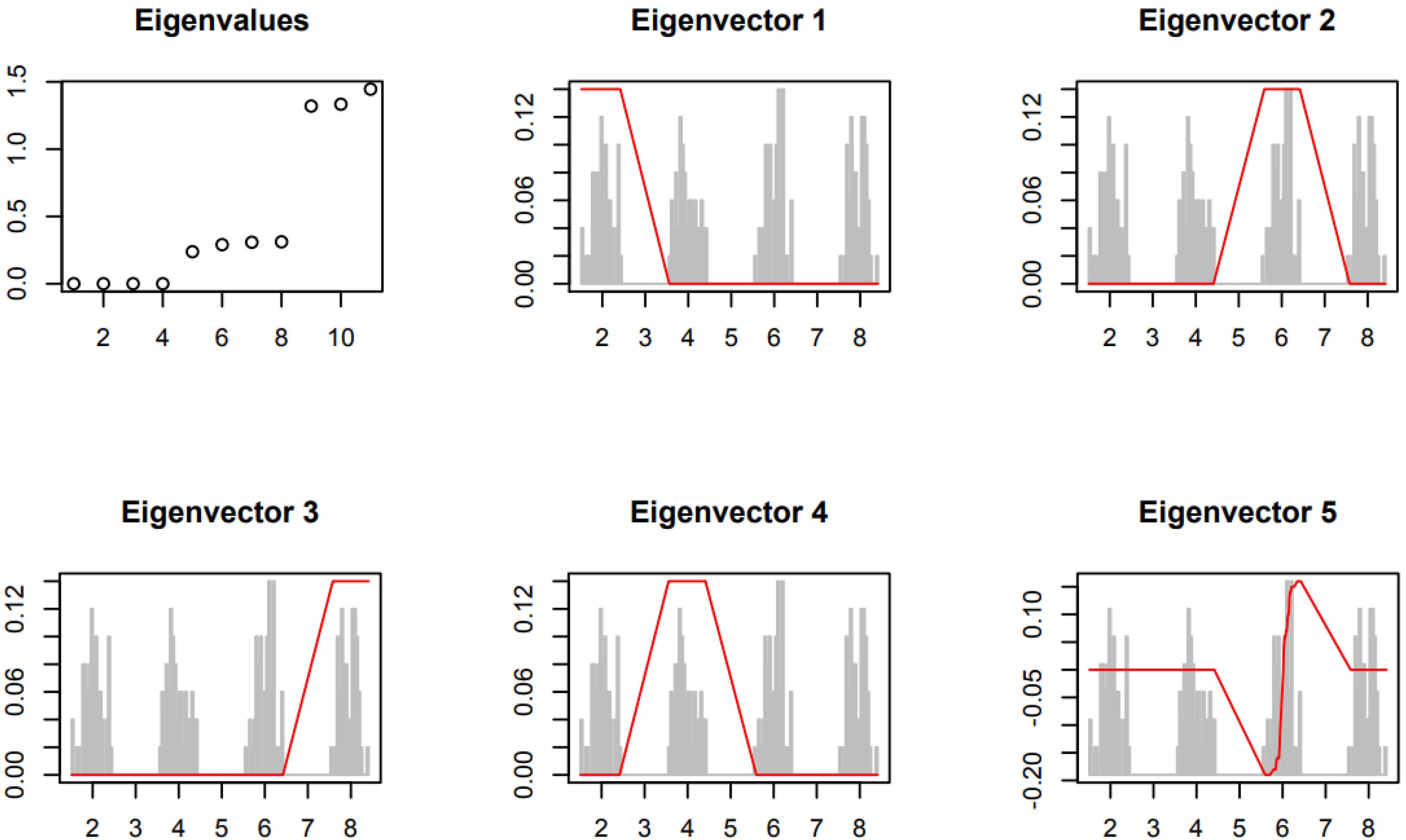
Output: Clusters $A_1, \ldots, A_k$ with $A_i = \{j \mid y_j \in C_i\}$.



Histogram of the sample
Eigenvalues
Eigenvector 1
Eigenvector 2
Eigenvector 3
Eigenvector 4
Eigenvector 5
norm, knn
unnorm, knn
norm, full graph
unnorm, full graph


Histogram of the sample
0
2
4
6
2
3
4
5
6
7
8


1 7 15 24 33 42 51 60 69 78 87 96 106 117 128 139 150 161 172 183 194



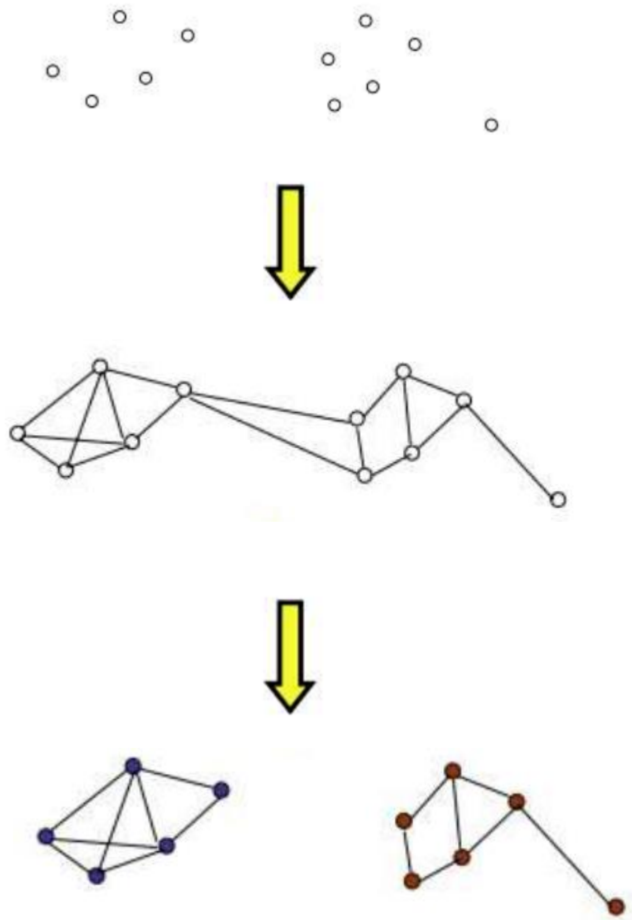
Eigenvalues
Eigenvector 1
Eigenvector 2
Eigenvector 3
Eigenvector 4
Eigenvector 5

- Brief Clustering Review
- Similarity Graph
- Graph Laplacian
- Spectral Clustering Algorithm
- Graph Cut Point of View
- Practical Details

Graph Cut Point of View

First - graph representation of data
(largely, application dependent)

Then - graph partitioning



Disconnected graph components ⟷ Groups of points(Weakly connections in between components Strongly connections within components)

$G=(V,E)$:

- For two not necessarily disjoint set $A, B \subset V$, we define

$$W(A,B) := \sum_{i\in A, j\in B} w_{ij}$$

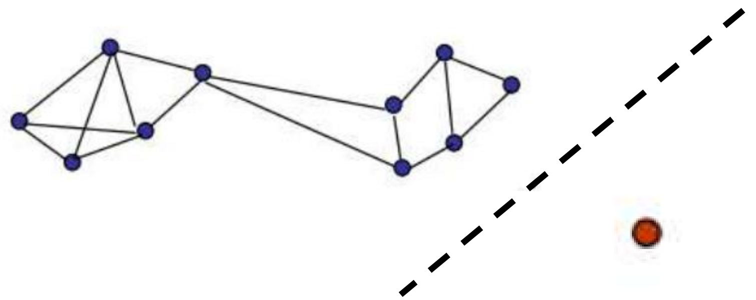
- Minicut: choosing a partition $A_1, A_2, \dots, A_K$ which minimizes

$$cut(A_1,\dots,A_k) := \frac{1}{2}\sum_{i=1}^{k} W(A_i, \overline{A_i})$$

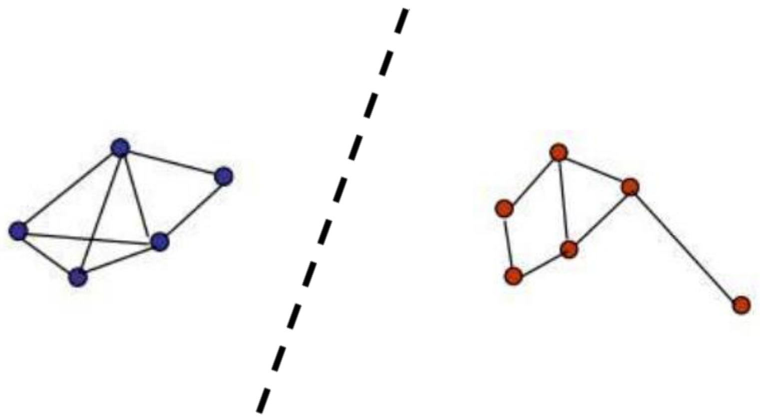
Cut between 2 sets $\quad cut(A_1, A_2) = \sum_{n\in A_1}\sum_{m\in A_2} w_{nm}$

Problems!!!

- Sensitive to outliers



What we get



What we want

电子科技大学

SMILE LAB

Statistical Machine Intelligence & Learning

Solutions

$|A|$:= the number of vertices in A

$$vol(A) := \sum_{i \in A} d_i$$

- RatioCut(Hagen and Kahng, 1992)

$$RatioCut(A_1, \dots, A_k) := \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{|A_i|} = \sum_{i=1}^{k} \frac{cut(A_i, \overline{A_i})}{|A_i|}$$

- Ncut(Shi and Malik, 2000)

$$Ncut(A_1, \dots, A_k) := \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{vol(A_i)} = \sum_{i=1}^{k} \frac{cut(A_i, \overline{A_i})}{vol(A_i)}$$

Problem!!!

- NP hard

$$RatioCut(A_1, \dots, A_k) := \frac{1}{2}\sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{|A_i|} = \sum_{i=1}^{k} \frac{cut(A_i, \overline{A_i})}{|A_i|}$$

$$Ncut(A_1, \dots, A_k) := \frac{1}{2}\sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{vol(A_i)} = \sum_{i=1}^{k} \frac{cut(A_i, \overline{A_i})}{vol(A_i)}$$

Solution!!!

- Approximation

Approximation $\rightarrow$ Spectral Clustering

Spectral Clustering $\rightarrow$ Ncut $\xrightarrow{\text{relaxing}}$ Normalized Spectral Clustering

Spectral Clustering $\rightarrow$ RatioCut $\xrightarrow{\text{relaxing}}$ Unnormalized Spectral Clustering

- **Approximation RatioCut for k=2**

Our goal is to solve the optimization problem:

$$\min_{A \subset V} RatioCut(A, \bar{A})$$

We first rewrite the problem in a more convenient form. Given a subset $A \subset V$ we define the vector $f = (f_1, \ldots, f_n)' \in \mathbb{R}^n$ with entries

$$f_i = \begin{cases} \sqrt{|\overline{A}|/|A|} & \text{if } v_i \in A \\ -\sqrt{|A|/|\overline{A}|} & \text{if } v_i \in \overline{A}. \end{cases}$$

- **Approximation RatioCut for k=2**

$$f_i = \begin{cases} \sqrt{|\bar{A}|/|A|}, & \text{if } v_i \in A \\ -\sqrt{|\bar{A}|/|A|}, & \text{if } v_i \in \bar{A} \end{cases}$$

$$RatioCut(A_1, \dots, A_k) := \frac{1}{2}\sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{|A_i|} = \sum_{i=1}^{k} \frac{cut(A_i, \overline{A_i})}{|A_i|}$$

$$cut(A_1, \dots, A_k) := \frac{1}{2}\sum_{i=1}^{k} W(A_i, \overline{A_i}) \qquad W(A, B) := \sum_{i \in A, j \in B} w_{ij}$$

$$f'Lf = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2$$

$$= \frac{1}{2}\sum_{i \in A, j \in \overline{A}} w_{ij}\left(\sqrt{\frac{|\overline{A}|}{|A|}} + \sqrt{\frac{|A|}{|\overline{A}|}}\right)^2 + \frac{1}{2}\sum_{i \in \overline{A}, j \in A} w_{ij}\left(-\sqrt{\frac{|\overline{A}|}{|A|}} - \sqrt{\frac{|A|}{|\overline{A}|}}\right)^2$$

$$= \mathrm{cut}(A, \overline{A})\left(\frac{|\overline{A}|}{|A|} + \frac{|A|}{|\overline{A}|} + 2\right)$$

$$= \mathrm{cut}(A, \overline{A})\left(\frac{|A| + |\overline{A}|}{|A|} + \frac{|A| + |\overline{A}|}{|\overline{A}|}\right)$$

$$= |V| \cdot \mathrm{RatioCut}(A, \overline{A}).$$

- Approximation RatioCut for k=2

Additionally, we have

$$\sum_{i=1}^{n} f_i = \sum_{i \in A} \sqrt{\frac{|\overline{A}|}{|A|}} - \sum_{i \in \overline{A}} \sqrt{\frac{|A|}{|\overline{A}|}} = |A| \sqrt{\frac{|\overline{A}|}{|A|}} - |\overline{A}| \sqrt{\frac{|A|}{|\overline{A}|}} = 0$$

The vector f as defined before is orthogonal to the constant one vector $\mathbb{1}$.

f satisfies

$$\|f\|^2 = \sum_{i=1}^{n} f_i^2 = |A| \frac{|\overline{A}|}{|A|} + |\overline{A}| \frac{|A|}{|\overline{A}|} = |\overline{A}| + |A| = n.$$

- Approximation RatioCut for k=2

$$f_i = \begin{cases} \sqrt{|\overline{A}|/|A|} & \text{if } v_i \in A \\ -\sqrt{|A|/|\overline{A}|} & \text{if } v_i \in \overline{A}. \end{cases}$$

$$\min_{A \subset V} \operatorname{RatioCut}(A, \overline{A}).$$

$$f'Lf = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2$$
$$= |V| \cdot \operatorname{RatioCut}(A, \overline{A}).$$

$$\min_{A \subset V} f'Lf \text{ subject to } f \perp \mathbb{1} \quad \|f\| = \sqrt{n}.$$

Relaxation !!!

$$\min_{f \in \mathbb{R}^n} f'Lf \text{ subject to } f \perp \mathbb{1}, \ \|f\| = \sqrt{n}.$$

Rayleigh-Ritz Theorem

f is the eigenvector corresponding to the second smallest eigenvalue of L (the smallest eigenvalue of L is 0 with eigenvector $\mathbb{1}$)

- Approximation RatioCut for k=2

f is the eigenvector corresponding to the second smallest eigenvalue of L

re-convert

Use the sign as indicator function

$$\begin{cases} v_i \in A & \text{if } f_i \geq 0 \\ v_i \in \overline{A} & \text{if } f_i < 0. \end{cases}$$

Only works for k = 2

f_i as points in R and do K-means

$$\begin{cases} v_i \in A & \text{if } f_i \in C \\ v_i \in \overline{A} & \text{if } f_i \in \overline{C}. \end{cases}$$

More General, works for any k

- **Approximation RatioCut for arbitrary k**

Given a partition of V into k sets $A_1, A_2, \dots, A_k$, we define k indicator vectors $h_j = (h_{1,j}, \dots, h_{n,j})'$ by

$$h_{i,j} = \begin{cases} \dfrac{1}{\sqrt{|A_j|}}, & \text{if } v_i \in A_j \\ 0, & \text{otherwise} \end{cases} \qquad (i=1,\dots,n;\ j=1,\dots,k)$$

$$h_i' L h_i = \frac{cut(A_i, \overline{A_i})}{|A_i|}$$

$$h_i' L h_i = (H' L H)_{ii}$$

$$\text{RatioCut}(A_1, \dots, A_k) = \sum_{i=1}^{k} h_i' L h_i = \sum_{i=1}^{k} (H' L H)_{ii} = \text{Tr}(H' L H),$$

$H \in \mathbb{R}^{n \times k}$, containing those k Indicator vectors as columns. Columns in H are orthonormal to each other, that is $H'H = I$

• Approximation RatioCut for arbitrary k

Problem reformulation:

$$h_{i,j} = \begin{cases} \frac{1}{\sqrt{|A_j|}}, & \text{if } v_i \in A_j \\ 0, & \text{otherwise} \end{cases}$$

$$\text{minimizing} \quad \text{RatioCut}(A_1, \dots, A_k)$$

$$\Updownarrow$$

$$\min_{A_1,\dots,A_k} Tr(H'LH) \ \textit{subject to}\ H'H = I$$

$\Downarrow$ **Relaxation !!!**

$$\min_{H \in \mathbb{R}^{n\times k}} Tr(H'LH) \ \textit{subject to}\ H'H = I$$

$\Downarrow$ **Rayleigh-Ritz Theorem**

Optimal H is the first k eigenvectors of L as columns.

- Brief Clustering Review
- Similarity Graph
- Graph Laplacian
- Spectral Clustering Algorithm
- Graph Cut Point of View
- Practical Details

PRACTICAL DETAILS

- Constructing the similarity graph

1. **Similarity Function Itself** Make sure that points which are considered to be "very similar" by the similarity function are also closely related in the application the data comes from.
2. **Type of Similarity Graph** Which one to choose from those three types. General recommendation: k-nearest neighbor graph.
3. **Parameters of Similarity Graph**(k or ε)
    1. KNN: k in order of log(n);
    2. mutual KNN: k significantly larger than standard KNN;
    3. ε-neighborhood graph: longest edge of MST;
    4. fully connected graph: σ in order of the mean distance of a point to its k-th nearest neighbor. Or choose $k = \varepsilon$.

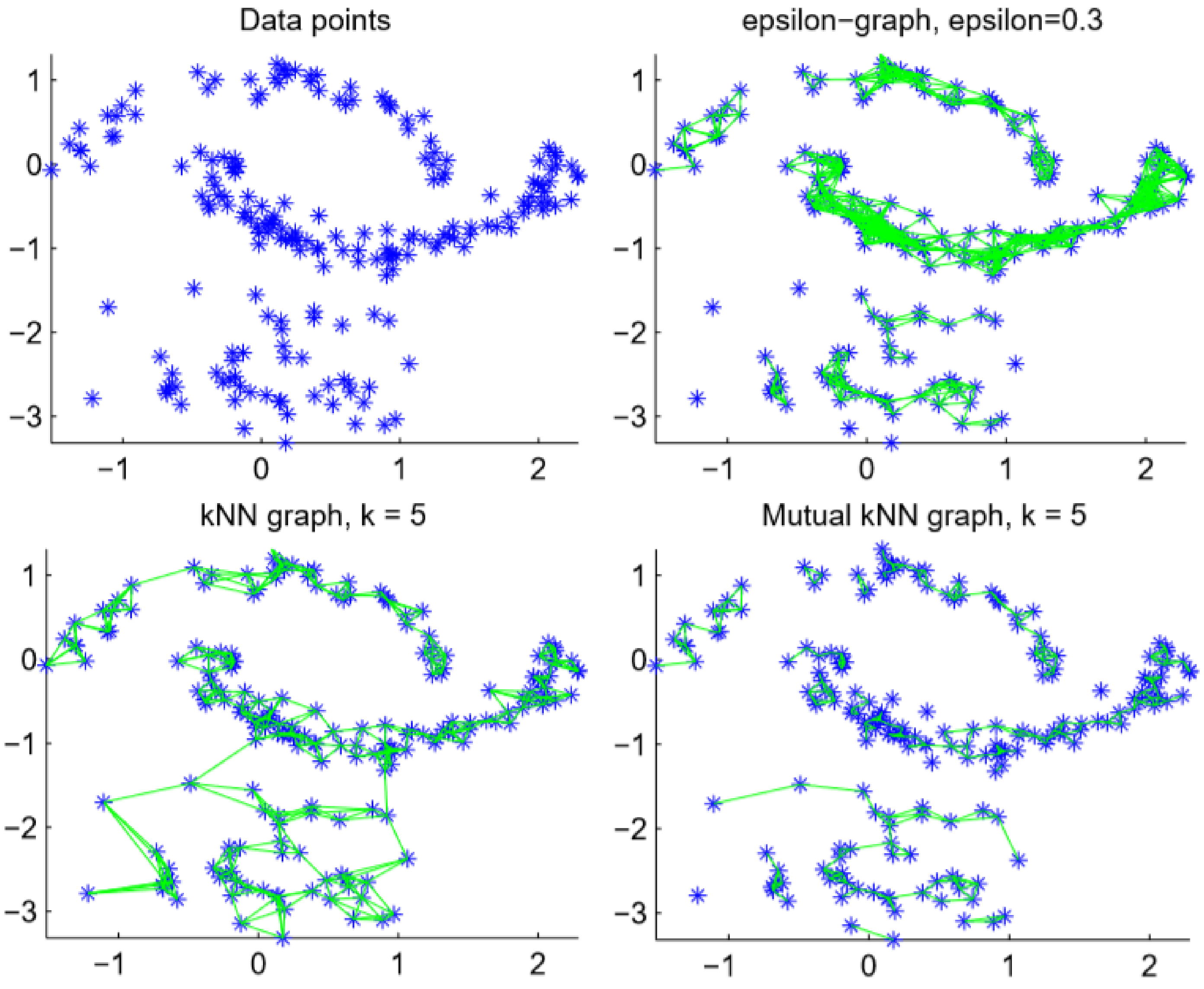
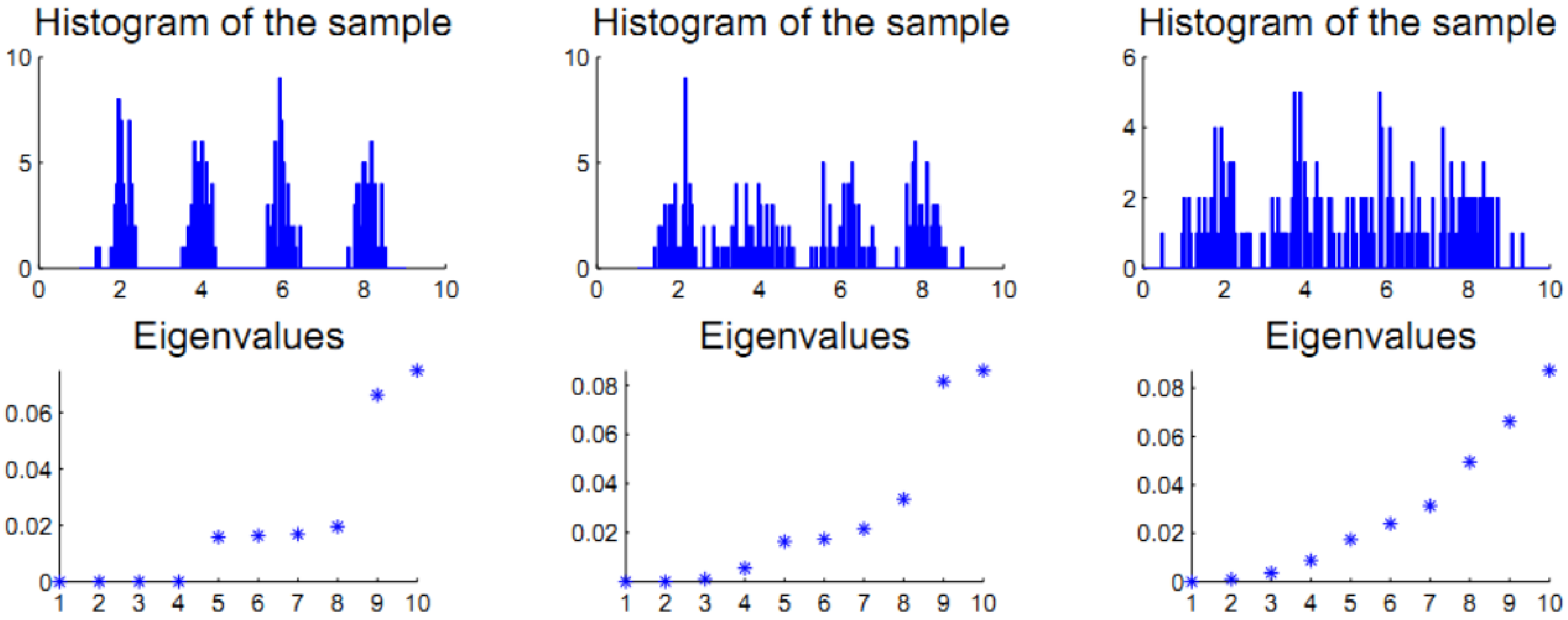


Figure 3: Different similarity graphs, see text for details.

- Computing Eigenvectors
    1. How to compute the first eigenvectors efficiently for large L
    2. Numerical eigensolvers converge to some orthonormal basis of the eigenspace.
- Number of Clusters
    1. General Strategies
    2. Eigengap heuristic(Choose the number k such that all eigenvalues $\lambda_1, \dots \lambda_k$ are very small, but λ_{k+1} is relatively large)



Well Separated | More Blurry | Overlap So Much

Eigengap Heuristic usually works well if the data contains very well pronounced clusters, but in ambiguous cases it also returns ambiguous results.

Spectral clustering

1. Does not make strong assumptions on the form of the clusters ⇒ can solve very general problems like intertwined spirals

2.can be implemented efficiently even for large data sets (as the adjacency matrix is sparse)

3.no issues of getting stuck in local minima or restarting the algorithm for several times with different initializations

but

1.choosing a good similarity graph is not trivial

2.spectral clustering can be quite unstable under different choices of the parameters for the similarity graph

Spectral clustering cannot serve as a "black box algorithm" which automatically detects the correct clusters in any given data set. But itcan be considered as a powerful tool which can produce good results if applied with care.

Thanks!