Università della Svizzera italiana

Institute of Computing CI

An introduction to graph clustering

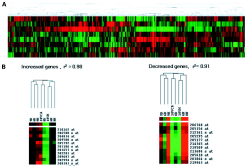
D. Pasadakis, L. Gaedke-Merzhäuser, E. Vecchi, and Prof. O. Schenk

November 2, 2022

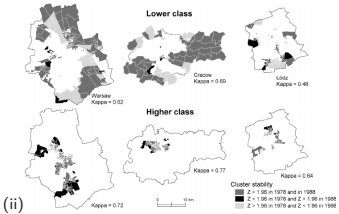
Motivation

Identify groups of similar behavior

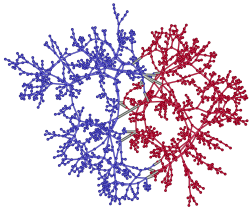
- Statistics
- Computer science
- Biology
- Social science and psychology
- ...



(i)



(ii)



i) Abhijit G. Banerjee, Indraneel Bhattacharyya, and Jamboor K. Vishwanatha. Identification of genes and molecular pathways involved in the progression of premalignant oral epithelia.Molecular Cancer Therapeutics, 4(6):865–875, 2005.
ii) Szymon Marcińczak, Michael Gentile, and Marcin Stępniak. Paradoxes of (post)socialist segregation: Metropolitan sociospatial divisions under socialism and after in Poland. Urban Geography, 34(3):327–352, 2013.

Overview

Clustering algorithms

- Flat algorithms
  - No explicit structure that would relate clusters to each other
  - Usually start with a random partitioning
    - K-means clustering
    - Model based clustering
  - Spectral clustering
- Hierarchical algorithms
  - Bottom-up, agglomerative
  - Top-down, divisive

Martin Grandjean. A social network analysis of Twitter: Mapping the digital humanities community. Cogent Arts & Humanities, 3(1), apr 2016.

Overview

Clustering algorithms

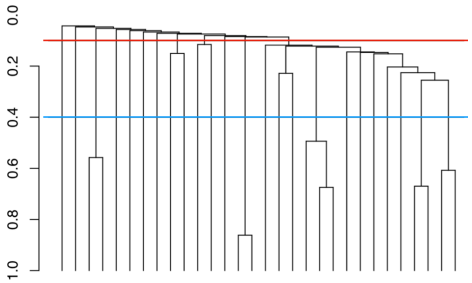
- Flat algorithms
  - No explicit structure that would relate clusters to each other
  - Usually start with a random partitioning
    - K-means clustering
    - Model based clustering
  - Spectral clustering
- Hierarchical algorithms
  - Bottom-up, agglomerative
  - Top-down, divisive

Martin Grandjean. A social network analysis of Twitter: Mapping the digital humanities community. Cogent Arts & Humanities, 3(1), apr 2016.

Hierarchical clustering

Agglomerative clustering ↑

- Agglomerate: to collect or gather into a cluster or mass.
- Treat each data entry as a singleton cluster, and then successively merge pairs of clusters until all clusters have been merged into a single one.
- Typically visualized as a dendrogram.
  - Horizontal line: cluster merge
  - Y-axis: the similarity of the two clusters
- Does not require a prespecified number of clusters.



Cut the hierarchy at:

- Large gap between two successive similarities $\Rightarrow$ natural clusters.
- Prespecified level of similarity.

Hierarchical clustering

Divisive clustering ↓

- Start at the top, with all data in one cluster.
- The cluster is split using a *flat* algorithm. Apply this procedure recursively.
- More complex (hierarchical + flat), but also more
  - * efficient if we don't generate a hierarchy all the way down, and
  - * accurate (in some circumstances), since it benefits from global information.

Hierarchical Pros & Cons

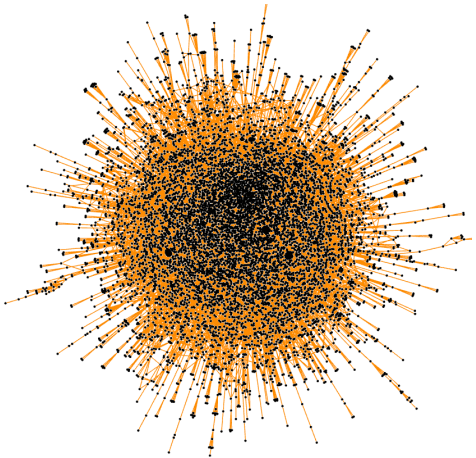
+ More informative than the unstructured set of flat clusters.
+ Easier to decide on the number of clusters by looking at the dendrogram.
– Impossible to undo the previous step: once the instances have been assigned to a cluster, they can no longer be moved around.
– Time complexity: at least $O(n^2)$, not suitable for very large datasets.
– The order of the data has an impact on the final results.

Introduction

Graph theory

can provide detailed information regarding the inner structure of a data set in terms of

* cliques – subsets of connected nodes, each pair of elements is connected,
* clusters – highly connected groups of nodes,
* centrality – important nodes, hubs, and
* outliers – unimportant nodes.



Introduction

Application domains - in literally every scientific field dealing with empirical data

NETWORK	NODES	LINKS	DIRECTED UNDIRECTED
Internet	Routers	Internet connections	Undirected
WWW	Webpages	Links	Directed
Power Grid	Power plants, transformers	Cables	Undirected
Mobile Phone Calls	Subscribers	Calls	Directed
Email	Email addresses	Emails	Directed
Science Collaboration	Scientists	Co-authorship	Undirected
Actor Network	Actors	Co-acting	Undirected
Citation Network	Paper	Citations	Directed
E. Coli Metabolism	Metabolites	Chemical reactions	Directed
Protein Interactions	Proteins	Binding interactions	Undirected

Albert-László Barabási and Márton Pósfai. Network science. Cambridge University Press, Cambridge, 20

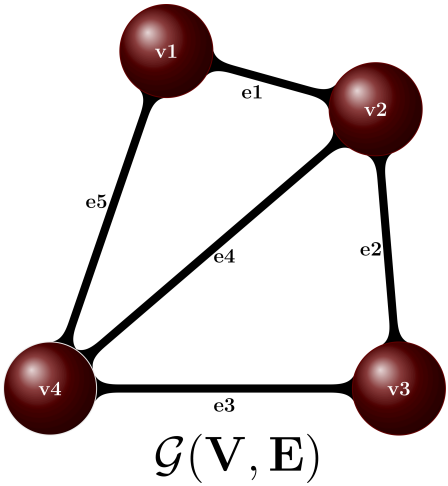
Graph notation

For an undirected graph $\mathcal{G}(V, E)$, with vertex set $V = \{v_1, \cdots, v_n\}$ and a subset $A \subset V$:

- Indicator vector: $\boldsymbol{x} \in \mathbb{R}^n$

$$x_i = \begin{cases} 1, & i \in A, \\ 0, & i \in \overline{A}. \end{cases}$$

- Shorthand notation: $i \in A = \{i | v_i \in A\}$.



Graph notation

For an undirected graph $\mathcal{G}(V, E)$, with vertex set $V = \{v_1, \cdots, v_n\}$:

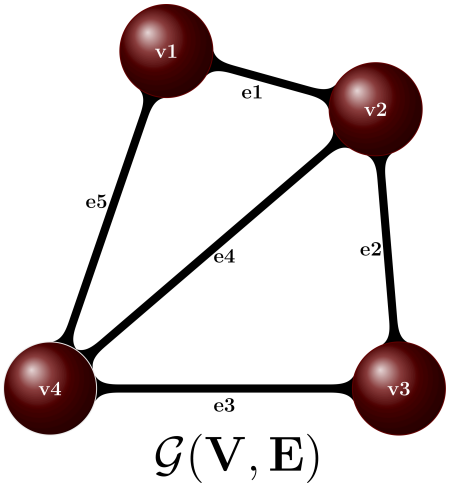
- Weights:

$$w_{ij} = \begin{cases} > 0, & \text{if } v_i \text{ connected to } v_j \\ = 0 & \text{otherwise.} \end{cases}$$

- Degree:

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

$\Rightarrow$ this sum only runs over all vertices adjacent to v_i. Why?

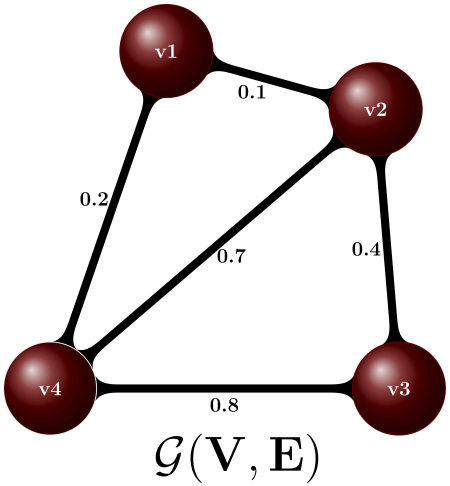


Graph notation

- Degree matrix : $\boldsymbol{D} \in \mathbb{R}^{n \times n}$

$$\boldsymbol{D} := \begin{bmatrix} d_1 & 0 & 0 & 0 \\ 0 & d_2 & 0 & 0 \\ 0 & 0 & d_3 & 0 \\ 0 & 0 & 0 & d_4 \end{bmatrix} \Rightarrow$$

$$\boldsymbol{D} = \begin{bmatrix} 0.3 & 0 & 0 & 0 \\ 0 & 1.2 & 0 & 0 \\ 0 & 0 & 1.2 & 0 \\ 0 & 0 & 0 & 1.7 \end{bmatrix}$$



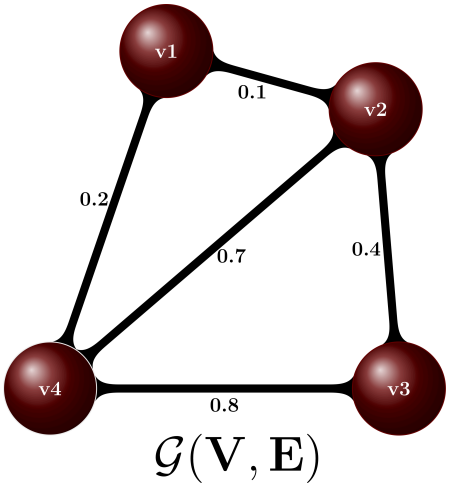
Graph notation

- For two sets $A, B \subset V$ we define the weighted adjacency matrix

$$\boldsymbol{W} := \sum_{i \in A, j \in B} w_{ij}, \quad \in \mathbb{R}^{n \times n}$$

$$\boldsymbol{W} := \begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} \\ w_{21} & w_{22} & w_{23} & w_{24} \\ w_{31} & w_{32} & w_{33} & w_{34} \\ w_{41} & w_{42} & w_{43} & w_{44} \end{bmatrix} \Rightarrow$$

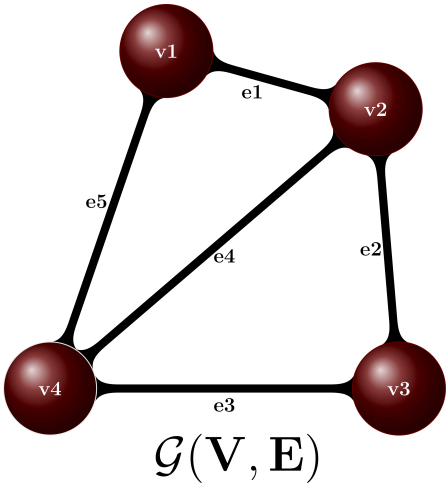
$$\boldsymbol{W} = \begin{bmatrix} 0 & 0.1 & 0 & 0.2 \\ 0.1 & 0 & 0.4 & 0.7 \\ 0 & 0.4 & 0 & 0.8 \\ 0.2 & 0.7 & 0.8 & 0 \end{bmatrix}$$



Graph notation

Measuring the size of a subset

1. $|A| :=$ the number of vertices in $|A|$
   cardinality, measures the size of A by its number of vertices.
2. $\text{vol}(A) := \sum_{i \in A} d_i$
   volume, measures the size of A by summing over the weights of all edges attached to vertices in A.



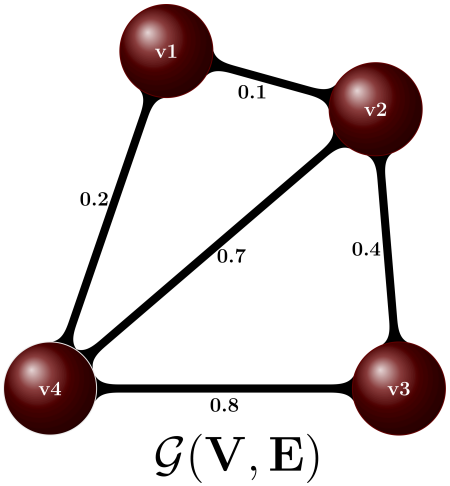
Graph notation

The unnormalized graph Laplacian matrix is defined as

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}.$$

$$\boldsymbol{L} := \begin{bmatrix} d_1 & -w_{12} & -w_{13} & -w_{14} \\ -w_{21} & d_2 & w_{23} & -w_{24} \\ -w_{31} & -w_{32} & d_3 & -w_{34} \\ -w_{41} & -w_{42} & -w_{43} & d_4 \end{bmatrix} \Rightarrow$$

$$\boldsymbol{L} = \begin{bmatrix} 0.3 & -0.1 & 0 & -0.2 \\ -0.1 & 1.2 & -0.4 & -0.7 \\ 0 & -0.4 & 1.2 & -0.8 \\ -0.2 & -0.7 & -0.8 & 1.7 \end{bmatrix}$$



Properties of $\boldsymbol{L}$

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

$$\boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x} = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(x_i - x_j).$$

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

Graph Construction - connectivity matrix $\boldsymbol{G}$

Goal: model the local neighborhood relationships between the data points.

- ϵ-neighborhood graph
  - Identify a threshold value ϵ, and include edges if the distance between two nodes is smaller or equal to ϵ.
- k-nearest neighbors (k-nn)
  - Insert edges between a node and its k-nearest neighbors.
  - Each node is connected to (at least) k nodes.
- Fully connected
  - An edge between every pair of nodes.
  - This is a full graph!

Graph Construction - similarity matrix $\boldsymbol{S}$

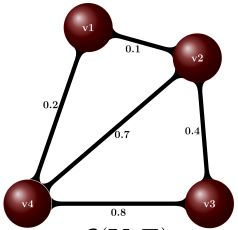
Goal: define a similarity function on the data. Ensure that the local neighborhoods promoted by this similarity function are "meaningful".

Example: in text documents check whether documents with a high similarity score indeed belong to the same text category.

Common case: data points in the Euclidean space $\mathbb{R}^d$ $\Rightarrow$ default candidate the Gaussian similarity function

$$s(x_i, x_j) = e^{\frac{-\left\|x_i - x_j\right\|^2}{2\sigma^2}}.$$

Graph Construction



Alltogether:

$$\boldsymbol{S} \odot \boldsymbol{G} = \boldsymbol{W}$$

$$\begin{bmatrix} 1 & 0.1 & 0.3 & 0.2 \\ 0.1 & 1 & 0.4 & 0.7 \\ 0.5 & 0.4 & 1 & 0.8 \\ 0.2 & 0.7 & 0.8 & 1 \end{bmatrix} \odot \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0.1 & 0 & 0.2 \\ 0.1 & 0 & 0.4 & 0.7 \\ 0 & 0.4 & 0 & 0.8 \\ 0.2 & 0.7 & 0.8 & 0 \end{bmatrix}$$

The $\boldsymbol{S}$ matrix represents the full similarity matrix constructed using the Gaussian similarity function. The $\boldsymbol{G}$ matrix represents an ϵ-similarity graph. The $\odot$ operator performs element-wise multiplication, resulting in a sparse matrix W, which only contains elements in places where $\boldsymbol{G}$ contains elements.

Graph Construction - choice of parameters I

Issue: If the similarity graph contains more connected components than the number of clusters we ask the algorithm to detect, the spectral clustering will trivially return connected components as clusters. Thus, the resulting graph should be connected, or at least have significantly fewer connected components than clusters we want to detect.

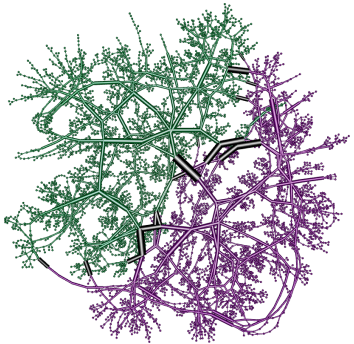
Lots of theoretical results on how connectivity of random graphs can be achieved, but all those results only hold in the limit for the sample size $n \to \infty$.

Rules of thumb

1. kNN graphs
   - small graphs: trial and error
   - big graphs: $k \sim \log(n)$
2. ϵ neighborhood graphs.
   Choose ϵ such that the resulting graph is safely connected, i.e the length of the longest edge in a minimal spanning tree of the fully connected graph on the data points.

Graph Construction - choice of parameters II

3. fully connected graph, from a Gaussian similarity function
   - choose σ in the order of the mean distance of a point to its k-th nearest neighbor, where $k \sim \log(n) + 1$
   - choose $\sigma = \epsilon$
   - all of these are ad-hoc

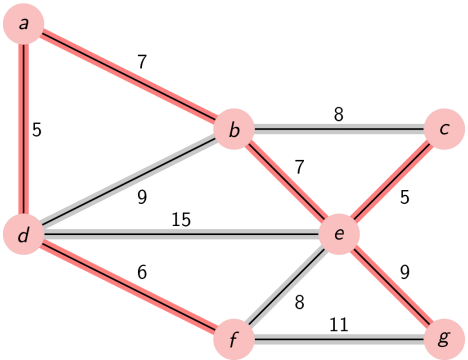


Spanning trees and minimal spanning trees

Tree: an undirected graph in which any two vertices are connected by exactly one path.

Spanning Tree: a subgraph that is a tree which includes all of the vertices of G, with minimum possible number of edges.

Minimum spanning tree: all of the above + minimum possible total edge weight (Prim '57).



Graph Construction

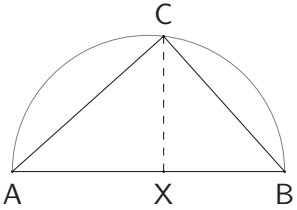
Algorithm 1 ϵ- similarity graph

Input: Pts, ϵ ▷ Number & threshold
Output: G ▷ The similarity matrix

```
1  function ϵ_Graph(Pts, ϵ)
2      for i=0:n do
3          for j=0:n do
4              dist = ||Pts(i) − Pts(j)||
5              if dist< ϵ then  ▷ Number & threshold
6                  G(i, j) = 1
7                  G(j, i) = 1
8              end if
9          end for
10     end for
11     return G
12 end function
```

Euclidean distance

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

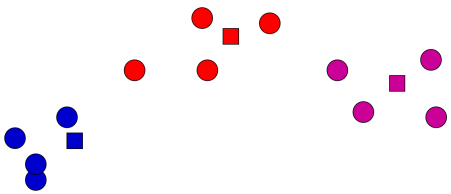


Graph Construction

Listing 1: kNN - similarity graph

```
for i = 1:n
    s = repeat(pts_spiral[[i],:],n,1);
    d = pts_spiral - s;
    e = sum(d.^2,dims=2);
    ind = sortperm(e);
    nbrs = ind[2:kn];
    G[i,nbrs] = ones(1,kn-1);
    G[nbrs,i] = ones(kn-1,1);
end
```

Example: 4-nearest neighbors



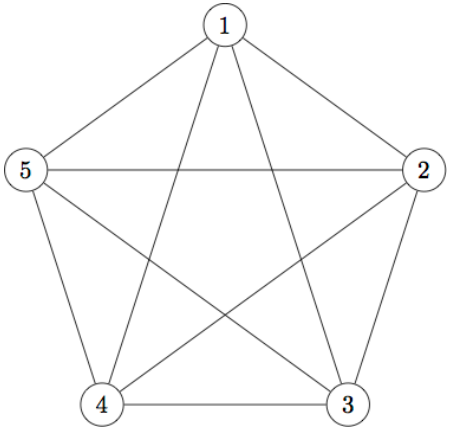
Graph Construction

Gaussian kernel similarity function

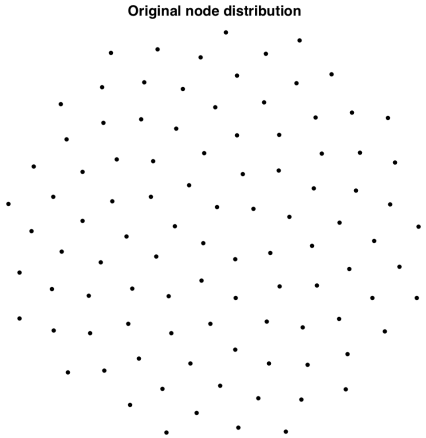
$$w_{ij} = e^{\frac{-\left\|x_i - x_j\right\|^2}{2\sigma^2}}$$

σ: controls the size of the neighborhood. $\Rightarrow$

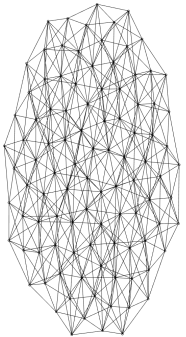
this leads to a complete graph (but with different weights)!

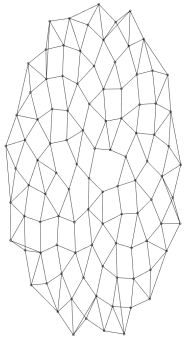


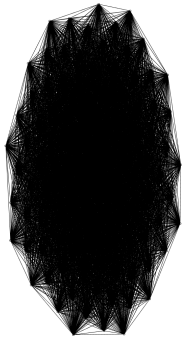
Graph Construction - Comparisons

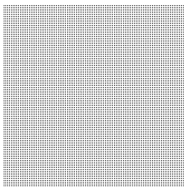


$n = 100$

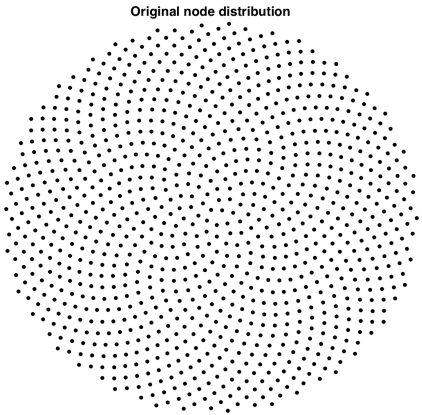




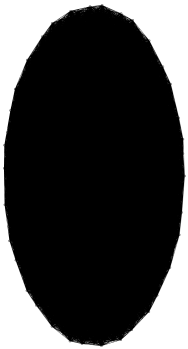


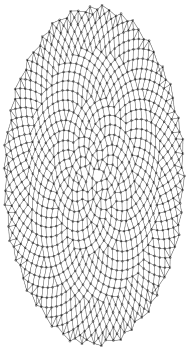


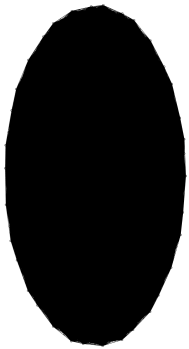
Graph Construction - Comparisons

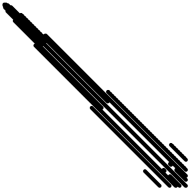


$n = 1000$





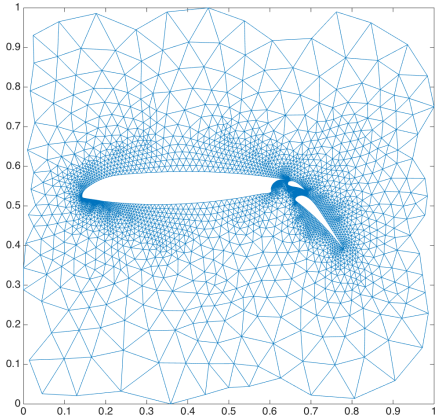




Drawing Graphs using Eigenvectors

A graph $\mathcal{G}(V, E)$ is a structure that models a relation E over a set of V entities. Relational information are visualized by graph drawing $\rightarrow$ usefulness depends on the clarity of the resulting layout.

```
W, xy = load("airfoil1.mat");
% matrix calculations
D = diagm(sum(W,dims=2));
L = D - W;

% Eigen-decomposition
lambda = eigvals(L);
Y = eigvecs(L);

ind = sortperm(lambda);
Y = Y[:, ind];

% Plot and compare
draw_graph(L, xy)
draw_graph(W, Y[:,2:3])
```



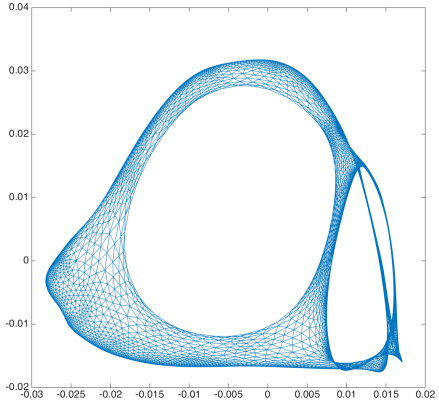
Drawing Graphs using Eigenvectors

Usually graphs don't come with coordinates, just with info on connectivity.

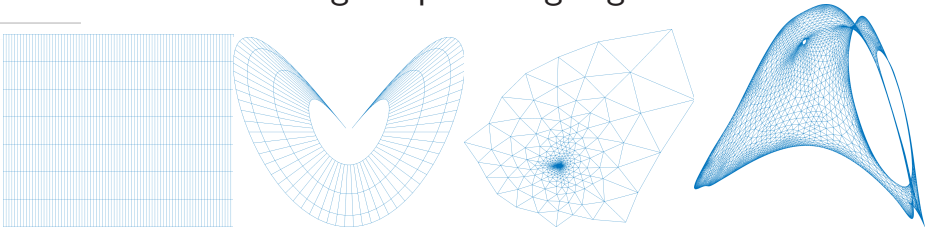
→ Draw them using the eigenvectors to supply coordinates. Locate vertex i at position:

$$x_i = (\boldsymbol{v}_2(i), \boldsymbol{v}_3(i))$$

```
W, xy = load("airfoil1.mat");
% matrix calculations
D = diagm(sum(W,dims=2));
L = D - W;

% Eigen-decomposition
lambda = eigvals(L);
Y = eigvecs(L);

ind = sortperm(lambda);
Y = Y[:, ind];

% Plot and compare
draw_graph(L, xy)
draw_graph(W, Y[:,2:3])
```



Drawing Graphs using Eigenvectors



Key advantages

1. Mathematically sound formulation, leading into an exact solution to the layout problem.
2. More visual information.
3. Fast computation.

Y. Koren. Drawing graphs by eigenvectors: theory and practice. Computers & Mathematics with Applications, 49(11):1867 – 1888, 2005.

K-Way spectral clustering

1. Recursive bi-partition (Hagen et al. '91)
   - Recursively apply bi-partitioning algorithm in a hierarchical divisive manner.
   - Advantage: Simple algorithmic idea, easy to realize by programming.
   - Disadvantages: Inefficient, unstable (Simon & Teng '97).

2. Cluster multiple eigenvectors (Shi & Malik '00)
   - Construct a reduced space from k eigenvectors.
   - Advantages: Make full use of the information of multiple eigenvectors; less computational complexity and the clustering results are quite satisfactory.
   - Disadvantages: Only the first four eigenvectors provide satisfactory results (Hendrickson & Leland, '95); not easy to select the appropriate number of eigenvectors (Jia '14).

Unnormalized spectral clustering

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

- Construct a similarity graph. Let $\boldsymbol{W}$ be its weighted adjacency matrix.
- Compute the unnormalized Laplacian $\boldsymbol{L}$.
- Compute the first k eigenvectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k$ of $\boldsymbol{L}$.
- Let $\boldsymbol{U} \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k$ as columns.

	$\boldsymbol{u}_1$	$\ldots$	$\boldsymbol{u}_k$
$\boldsymbol{U}_1$	u_{11}	$\ldots$	u_{1k}
$\vdots$	$\vdots$	$\ldots$	$\vdots$
$\boldsymbol{U}_n$	u_{n1}	$\ldots$	u_{nk}

  $\Rightarrow$ Dimensionality reduction: $n \times n \to n \times k$

- For $i = 1, \ldots, n$ let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of $\boldsymbol{U}$.
- 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

Interlude ☕

Overviews of clustering routines

- Wierzchón ST, Klopotek MA (2018) Modern Algorithms of Cluster Analysis, Springer International Publishing, Cham. DOI 10.1007/978-3-319-69308-8

- Xu, D., Tian, Y. (2015) A Comprehensive Survey of Clustering Algorithms. Ann. Data. Sci. 2, 165–193 . DOI 10.1007/s40745-015-0040-1

K-means clustering

- Most well-known and popular clustering algorithm.
- Cluster n objects based on attributes into k partitions, where $k < n$. We'll call these vectors $\boldsymbol{x_1}, \ldots, \boldsymbol{x_n}$ and the clusters $C_1, \ldots, C_k$.
- Classify/group the objects based on attributes/features into k groups.

K-means

1. Start with some initial cluster centers.
2. Iterate:
   - Assign/cluster each example to closest center.
   - Recalculate centers as the mean of the points in a cluster.

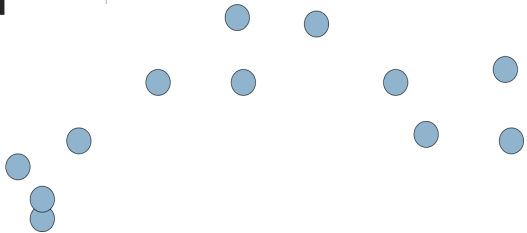
K-means

1. Start with some initial cluster centers.
2. Iterate:
   - Assign/cluster each example to closest center.
   - Recalculate centers as the mean of the points in a cluster.

K-means

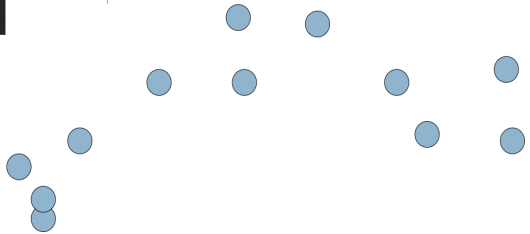
1. Start with some initial cluster centers.
2. Iterate:
   - Assign/cluster each example to closest center.
   - Recalculate centers as the mean of the points in a cluster.

K-means example

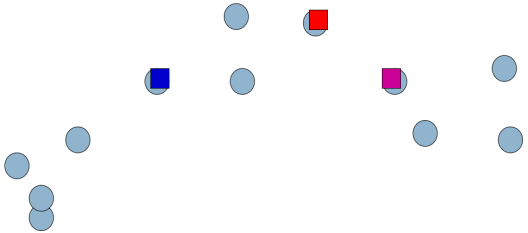


Initial distribution

K-means example

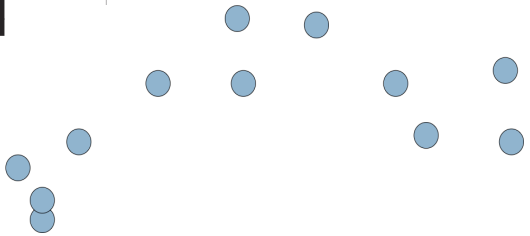


Initial distribution

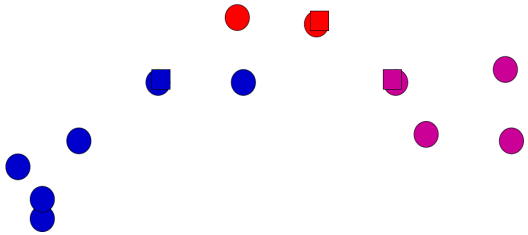


Initialize centers randomly

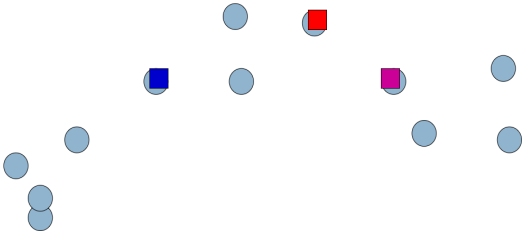
K-means example



Initial distribution

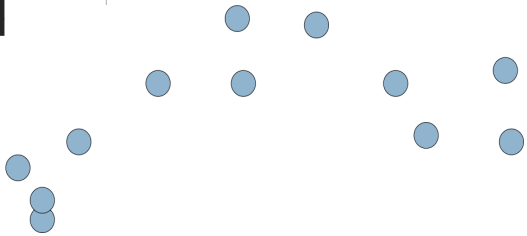


Assign points to nearest center

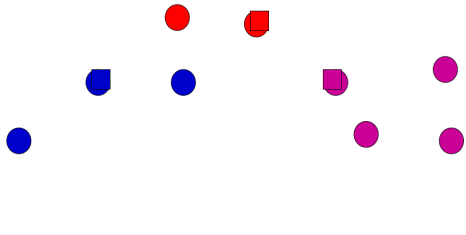


Initialize centers randomly

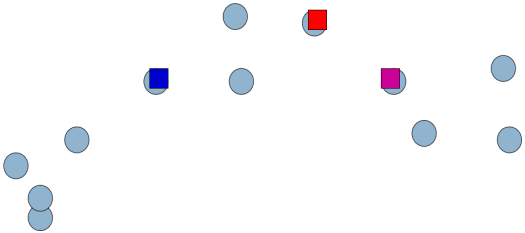
K-means example



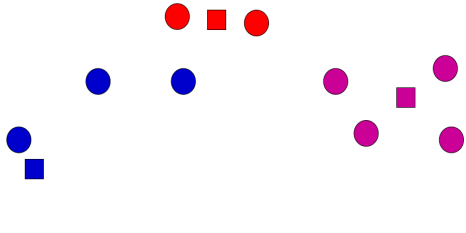
Initial distribution



Assign points to nearest center

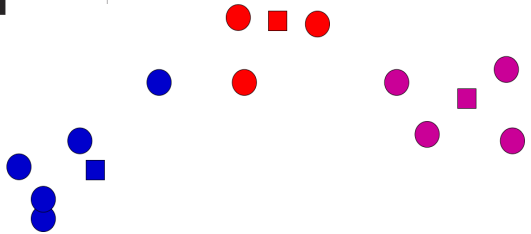


Initialize centers randomly



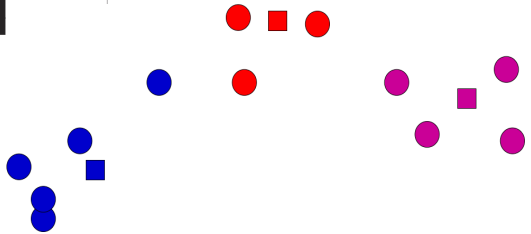
Readjust centers

K-means example

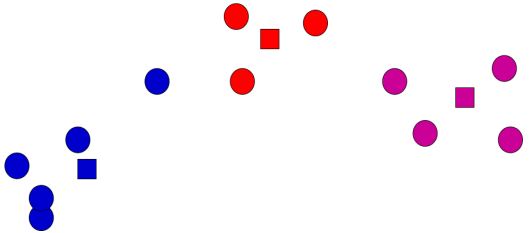


Assign points to nearest center

K-means example

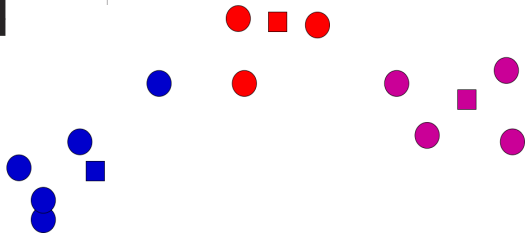


Assign points to nearest center

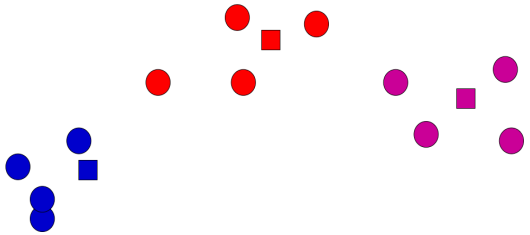


Readjust centers

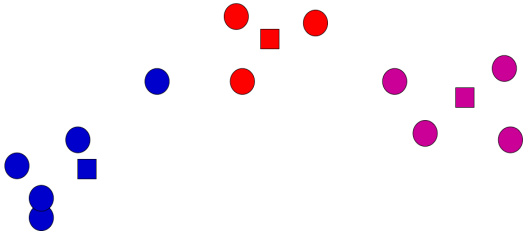
K-means example



Assign points to nearest center

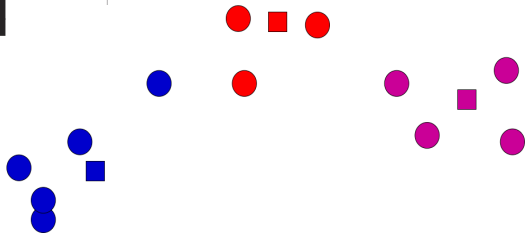


Assign points to nearest center

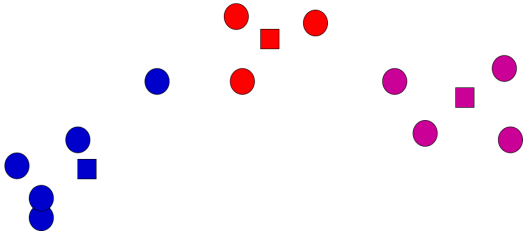


Readjust centers

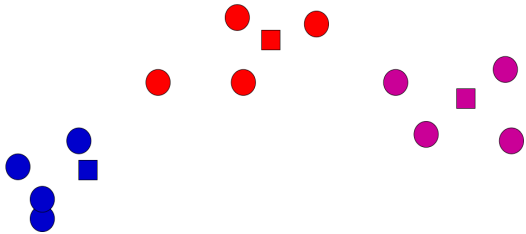
K-means example



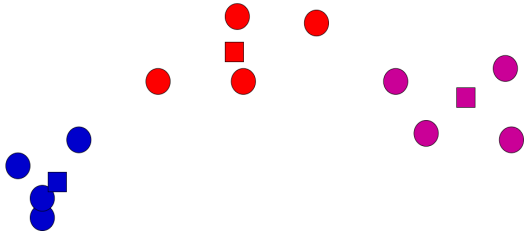
Assign points to nearest center



Readjust centers

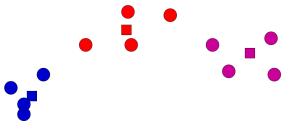


Assign points to nearest center



Readjust centers

K-means example



If no changes: DONE

Objective function:

$$\sum_{\alpha=1}^{k} \frac{1}{|C_\alpha|} \sum_{i,j \in C_\alpha} \|x_i - x_j\|$$

Set μ_α to be the average of the points in C_α

$$\sum_{\alpha=1}^{k} \sum_{i \in C_\alpha} \|x_i - \mu_\alpha\|^2$$

Algorithm by Lloyd '82. Alternating steps, where the cluster-averages $\mu_1, \ldots, \mu_k$ are computed, then each point is shifted to the cluster with the closest center.

1. $\forall\ 1 \leq \alpha \leq k$, set

$$\mu_\alpha = \frac{1}{|C_\alpha|} \sum_{i \in C_\alpha} x_i.$$

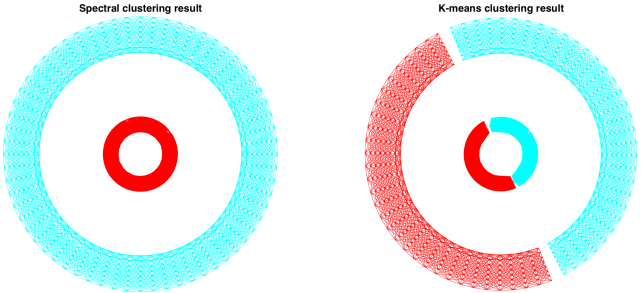
2. $\forall\ 1 \leq i \leq n$, put i in the cluster α for which

$$\|\mu_\alpha - x_i\|$$

is lowest.

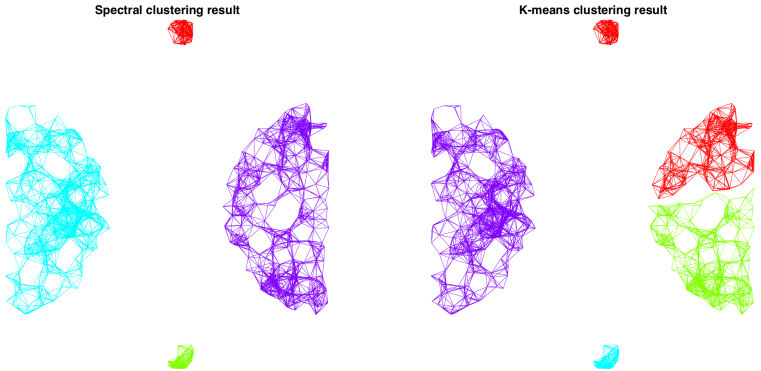
K-means drawbacks - I

1. Use of Euclidean distance ⇒ data space treated as isotropic (distances unchanged by translations and rotations). The data points in each cluster are modeled as lying within a sphere around the cluster centroid. The clusters are modeled only by the position of their centroids, so K-means implicitly assumes all clusters have the same radius. When this implicit equal-radius, spherical assumption is violated, K-means can behave in a non-intuitive way, even when clusters are very clearly identifiable.



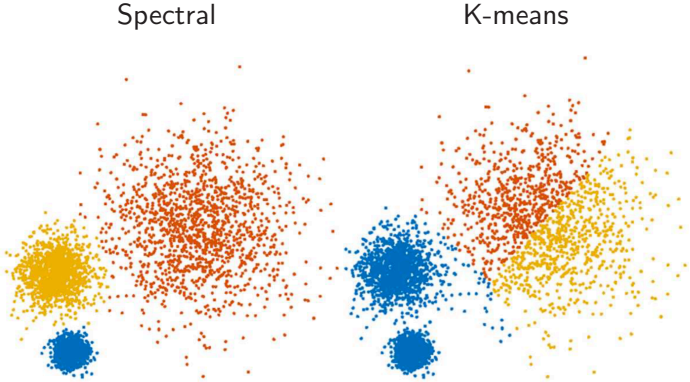
K-means drawbacks - II

❷ Euclidean space is linear ⇒ small changes in the data result in proportionately small changes to the position of the cluster centroids. This is problematic when there are outliers, i.e points which are unusually far away from the cluster centroid by comparison to the rest of the points in that cluster.



K-means drawbacks - III

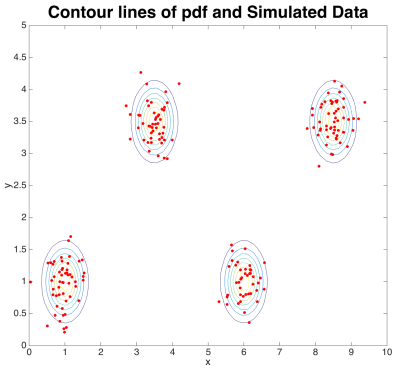
3. K-means clusters data points purely on their (Euclidean) geometric closeness to the cluster centroid $\Rightarrow$ does not take into account the different densities of each cluster. So, because K-means implicitly assumes each cluster occupies the same volume in data space, each cluster must contain the same number of data points.

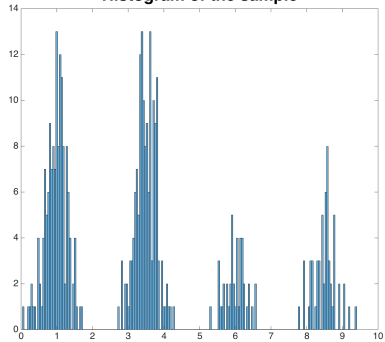


The spectrum of $\boldsymbol{L}$

Let G be an undirected graph with $w_{ij} \neq 0$. Then the multiplicity k of the eigenvalue 0 of $\boldsymbol{L}$ 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.

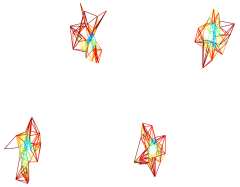
$\Rightarrow$ We will demonstrate this with the following example (as per Luxburg '07):

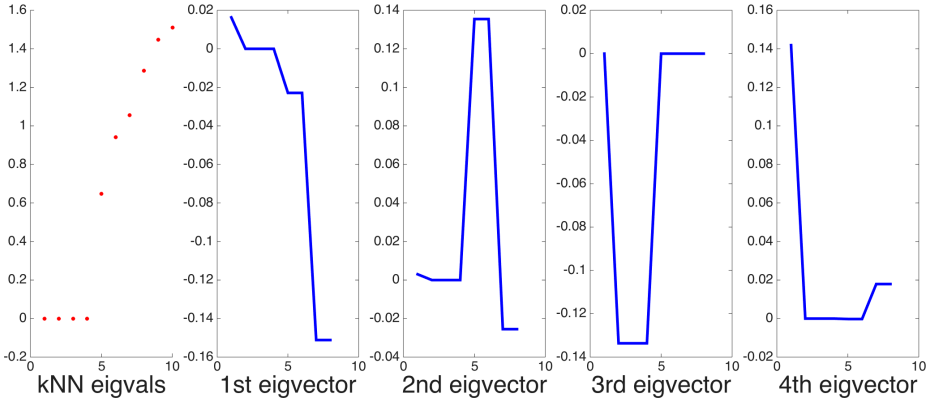




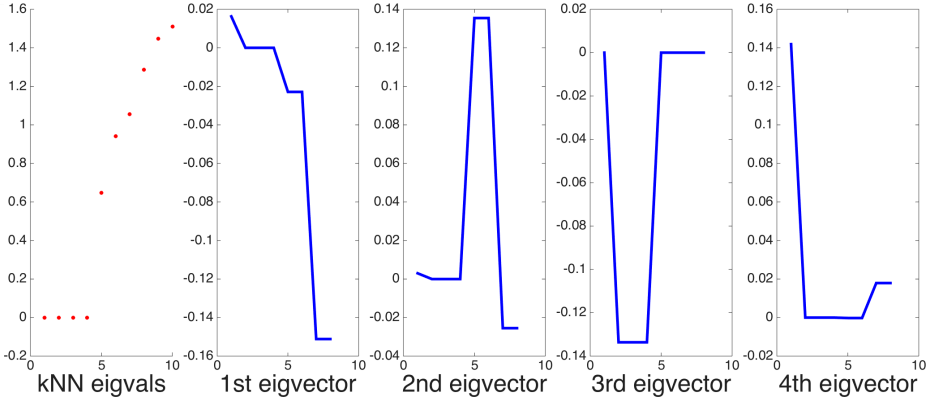
The kNN case

$\Rightarrow$ 10 neighbors





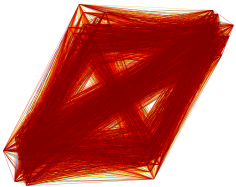
The kNN case

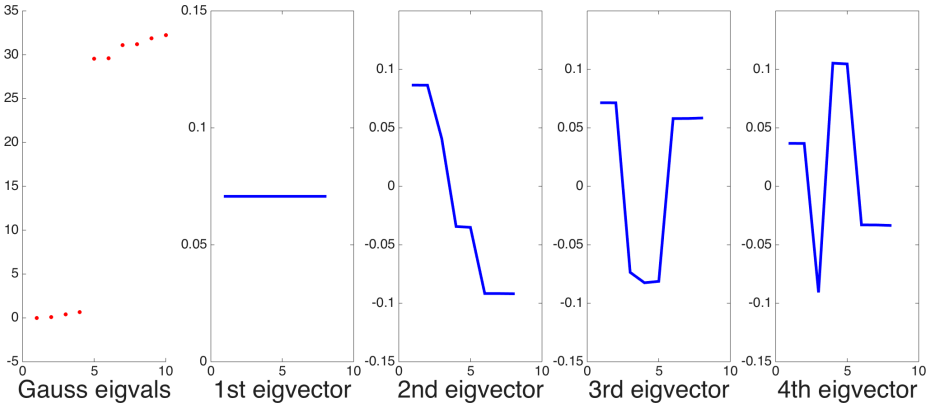


- The first four eigenvalues are 0, and the corresponding eigenvectors are cluster indicator vectors. The reason is that the clusters form disconnected parts in the 10-nearest neighbor graph.

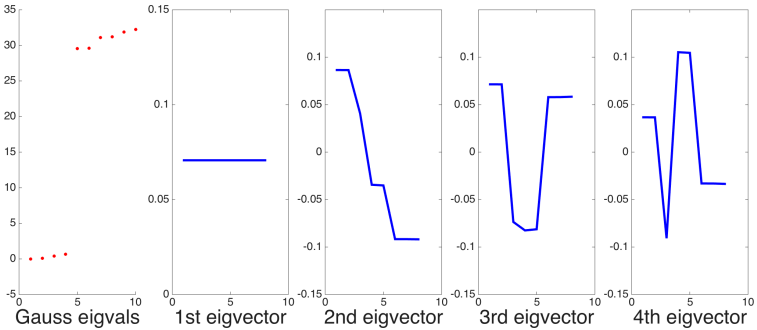
The full graph case

$\sigma = 1$





The full graph case



- This graph consists of only one connected component.
- Eigenvalue 0 has multiplicity 1, and the first eigenvector is the constant vector.
- Thresholding the 2nd eigenvector at 0, the part < 0 corresponds to clusters 1 and 2, and the part > 0 to clusters 3 and 4. Similarly, the 3rd eigenvector separates clusters 1 and 4 from clusters 2 and 3, and the 4th eigenvector separates clusters 1 and 3 from clusters 2 and 4. $\Rightarrow$ The first four eigenvectors carry all the information about the clusters.

Balanced graph partitioning analogy

Insight: separate points in different groups according to their similarities.

$\rightarrow$ Graph Theory $\leftarrow$

Analogy: partition a graph such that the edges between different groups have a very low weight (i.e points in different clusters are dissimilar) and edges within a group have high weight (points within the same cluster are similar).

Balanced graph partitioning analogy

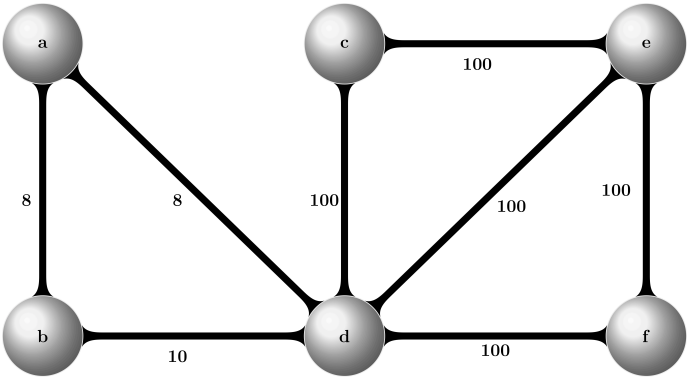
For a given number k of subsets, we want to find a partition $A_1, \cdots, A_k$ that minimizes

$$\text{cut}(A_1, \cdots, A_k) = \frac{1}{2} \sum_{i=1}^{k} W(A_i, \overline{A_i}) \quad \Rightarrow \quad \text{This is not balanced}$$

$$\text{RatioCut}(A_1, \cdots, A_k) = \sum_{i=1}^{k} \frac{\text{cut}(A_1, \cdots, A_k)}{|A_i|} \quad \Rightarrow \quad \text{This is } \text{😎}$$

The minimum of $\sum_{i=1}^{k} (1/|A_i|)$ is met if all $|A_i|$ coincide.

Balanced graph partitioning analogy



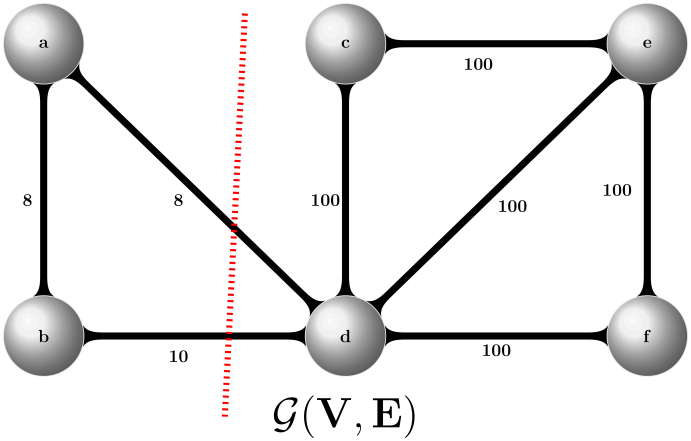
$\mathcal{G}(\mathbf{V}, \mathbf{E})$

minimum cut = ?

optimal bisection = ?

Ratio Cut = ?

Balanced graph partitioning analogy



$$\text{minimum cut} = \text{a}|\text{bcdef}$$

$$\text{optimal bisection} = \text{abc}|\text{def}$$

$$\text{Ratio Cut} = \text{ab}|\text{cdef} = 18 \times \left(\frac{1}{2} + \frac{1}{4}\right)$$

$$= 13.5$$

Approximating RatioCut

Let $k = 2$. Our goal is to solve the minimization problem

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

Given a subset $A \subset V$, we define the vector $\boldsymbol{x} = (x_1, \ldots, x_n)^\mathsf{T} \in \mathbb{R}^n$, with

$$x_i = \begin{cases} \sqrt{\frac{|\overline{A}|}{|A|}}, & \text{if } v_i \in A, \\ -\sqrt{\frac{|A|}{|\overline{A}|}}, & \text{if } v_i \in \overline{A}. \end{cases}$$

The RatioCut objective function can now be rewritten using the graph Laplacian.

$$\boldsymbol{x}^\mathsf{T} \boldsymbol{L} \boldsymbol{x} = \boldsymbol{x}^\mathsf{T} \boldsymbol{D} \boldsymbol{x} - \boldsymbol{x}^\mathsf{T} \boldsymbol{W} \boldsymbol{x} = \sum_{i=1}^{n} d_i x_i^2 - \sum_{i,j=1}^{n} x_i x_j w_{ij} = \frac{1}{2} \left(\sum_{i=1}^{n} d_i x_i^2 - 2 \sum_{i,j=1}^{n} x_i x_j w_{ij} + \sum_{j=1}^{n} d_j x_j^2 \right) = \ldots$$

Approximating RatioCut

$$
\begin{aligned}
\ldots &= \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(x_i - x_j)^2 \\
&= \frac{1}{2}\sum_{i\in A, j\in\overline{A}} w_{ij}\left(\sqrt{\frac{|\overline{A}|}{|A|}} + \sqrt{\frac{|A|}{|\overline{A}|}}\right)^2 + \frac{1}{2}\sum_{i\in\overline{A}, j\in A} w_{ij}\left(-\sqrt{\frac{|\overline{A}|}{|A|}} - \sqrt{\frac{|A|}{|\overline{A}|}}\right)^2 + 0 + 0 \\
&= \frac{1}{2}\sum_{i\in A, j\in\overline{A}} w_{ij}\left(\frac{|\overline{A}|}{|A|} + 2\sqrt{\frac{|\overline{A}|}{|A|}\frac{|A|}{|\overline{A}|}} + \frac{|A|}{|\overline{A}|}\right) + \frac{1}{2}\sum_{i\in\overline{A}, j\in A} w_{ij}\left(\frac{|\overline{A}|}{|A|} + 2\sqrt{\frac{|A|}{|\overline{A}|}\frac{|\overline{A}|}{|A|}} + \frac{|A|}{|\overline{A}|}\right) \\
&= \left(\frac{|\overline{A}|}{|A|} + \frac{|A|}{|\overline{A}|} + 2\right)\cdot\left(\frac{1}{2}\sum_{i\in A, j\in\overline{A}} w_{ij} + \frac{1}{2}\sum_{i\in\overline{A}, j\in A} w_{ij}\right) \\
&= \left(\frac{|\overline{A}| + |A|}{|\overline{A}|} + \frac{|A| + |\overline{A}|}{|A|}\right)\cdot \operatorname{cut}\left(A, \overline{A}\right) = \cdots
\end{aligned}
$$

Approximating RatioCut

$$\begin{aligned} \ldots &= \operatorname{cut}\left(A, \overline{A}\right) \cdot \left(\frac{1}{|A|} + \frac{1}{|\overline{A}|}\right) \cdot \left(|A| + |\overline{A}|\right) \\ &= \operatorname{RCut}\left(A, \overline{A}\right) \cdot |V|. \end{aligned}$$

Additionally, the vector $\boldsymbol{x}$ is orthogonal to the constant vector $\mathbb{1}$.

$$\boldsymbol{x}^{\mathsf{T}}\mathbb{1} = \sum_{i=1}^{n} x_i = \sum_{i \in A} \sqrt{\frac{|\overline{A}|}{|A|}} - \sum_{i \in \overline{A}} \sqrt{\frac{|A|}{|\overline{A}|}} = |A|\sqrt{\frac{|\overline{A}|}{|A|}} - |\overline{A}|\sqrt{\frac{|A|}{|\overline{A}|}} = 0.$$

And finally, $\boldsymbol{x}$ measures the cardinality of the entire set:

$$\|\boldsymbol{x}\|^2 = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{x} \sum_{i=1}^{n} x_i^2 = |A| \cdot \frac{|\overline{A}|}{|A|} + |\overline{A}| \cdot \frac{|A|}{|\overline{A}|} = |A| + |\overline{A}| = n = |V|.$$

Approximating RatioCut

Alltogether the minimization problem

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

can be equivalently stated as

$$\min_{A \subset V} \boldsymbol{x}^{\mathsf{T}} \boldsymbol{L} \boldsymbol{x} \text{ subject to } \boldsymbol{x} \perp \mathbb{1},$$

$$x_i = \begin{cases} \sqrt{\frac{|\overline{A}|}{|A|}}, & \text{if } v_i \in A, \\ -\sqrt{\frac{|A|}{|\overline{A}|}}, & \text{if } v_i \in \overline{A}, \end{cases}$$

$$\|\boldsymbol{x}\|^2 = n.$$

This is a discrete optimization problem, due to the entries of the solution vector $\boldsymbol{x}$, $\Rightarrow$ NP-hard.

Approximating RatioCut

Allowing x_i to attain value in $\mathbb{R}$, leads to the relaxed optimization problem

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \boldsymbol{x}^\mathsf{T} \boldsymbol{L} \boldsymbol{x} \text{ subject to } \boldsymbol{x} \perp \mathbb{1}, \|\boldsymbol{x}\|^2 = n.$$

The Rayleigh-Ritz theorem indicates that the solution of this problem is given by the Fiedler eigenvector (the eigenvector corresponding to the 2nd smallest eigenvalue of $\boldsymbol{L}$). However, in order to obtain a partition (cluster) of the graph we need to reverse the real-valued solution vector of the relaxed problem into a discrete indicator vector. In the case of spectral **partitioning**, we use the sign of $\boldsymbol{x}$ as an indicator function,

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

Approximating RatioCut

What spectral **clustering** algorithms do instead, is to regard the coordinated x_i as points in $\mathbb{R}$, and cluster them into two groups C and $\overline{C}$ with a *flat* clustering algorithm (k-means in our case). The clustering results are then associated with the underlying data points, that is we choose

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

This procedure is *unnormalized spectral clustering* for k $= 2$ clusters.

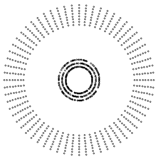
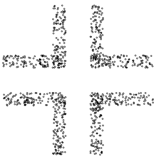
Computational complexity

1. Computation of eigenvalues/eigenvectors
   (Most expensive step)
   $$O(n^3)$$
2. Construction of similarity matrix
   (full graph case)
   $$O(n^2)$$
3. k-means step
   $$O(nldk)$$

n: number of nodes, l: number of k-means iterations, d: dimensionality of the input data, k: number of final clusters

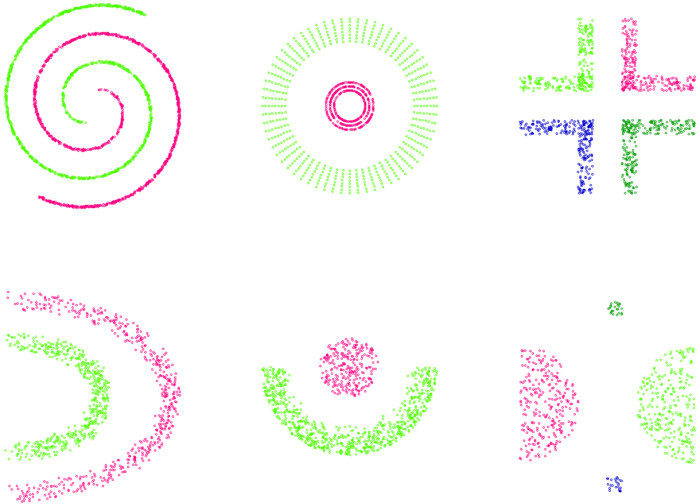
Showcase: Comparison with *k*-means

The following sets are good candidates to illustrate their differences.


Showcase: Comparison with k-means

We will try to recover the following clusters.



Take-home messages

- Clustering can be formulated as a graph cut problem.
- Effective in non-convex clustering scenarios.
- The min-cut problem can be solved by an eigen-decomposition of the Laplacian matrix.
- We obtain a data representation in a lower dimensional space $\rightarrow$ easily clustered.
- Empirically very successful.

Complementary reading


Hongjie Jia, Shifei Ding, Xinzheng Xu, and Ru Nie.
The latest research progress on spectral clustering.
Neural Comput. Appl., 24(7-8):1477–1486, June 2014.


Ulrike Luxburg.
A tutorial on spectral clustering.
Statistics and Computing, 17(4):395–416, December 2007.


Mark Newman.
Networks: An Introduction.
Oxford University Press, Inc., New York, NY, USA, 2010.

Complementary reading

- von Luxburg '07: Chapters 1, 2, 3.1, 4 (only the first algorithm), 5.1, 8.1, 8.2, 8.3, 8.4.
- Jia '14: Basic concepts and open research questions.
- Newman '10: Chapters 2-5 for an overview of different graph application domains, Chapters 6.1-6.3 and 6.13 for more graph notation, Chapters 7.1,7.12 for some measures and metrics, Chapters 11.1-11.6 for graph partitioning and graph clustering.
- http://scikit-learn.org/ - Overview of various clustering method