Notes on Statistical and Machine Learning

Clustering Analysis

Chapter: *22* **Prepared by:** *Chenxi Zhou*

This note is produced based on

- *Chapter 12, Clustering Analysis* in Izenman (2009),
- *Chapter 14, Unsupervised Learning* in Hastie, Tibshirani, and Friedman (2009),
- *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise* by Ester et al. (1996), and
- *A Tutorial on Spectral Clustering* by Luxburg (2007).

I. Introduction

1. **Introduction:** *Clustering analysis*, also known as *data segmentation* or *class discovery*, is an example of unsupervised learning and is a popular tool for analyzing unstructured multivariate data.
2. **Assumption:** The underlying assumption is that the data form a heterogenous set that should separate into natural groups familiar to the domain experts.

   Remark. In practice, there is no guarantee that more than one group can be found.
3. **Goal:** Its goal is to segment a given dataset into homogenous subgroups or "clusters".
4. **Cluster:** A *cluster* is generally thought of as a group of items satisfying the following properties:

   (a) each item within a cluster is "close" (in some appropriate sense) to a central item of a cluster, and

   (b) items of different clusters are "far away" from each other.

   In particular, methods for clustering items depend on how similar or dissimilar the items are to each other.
5. **Clustering Tasks:**

   (a) *Clustering Observations:* We cluster n observations into groups, where the number of groups, denoted by K, is unknown and has to be determined from the data.

   (b) *Clustering Variables:* We partition p variables into K distinct groups, where the number K is unknown.

- It is possible that one variable forms a cluster, but most clusters will be formed by several variables;
- These clusters should be *far enough apart* that groupings are easily identifiable;
- Each cluster of variables may be replaced by a single variable representative of that cluster.

(c) *Two-way Clustering:* We can cluster both observations and variables simultaneously.

In the remaining note, we focus on clustering observations only.

6. **Comparison with Classification:** Clustering and classification are different from the following perspectives:

   (a) in terms of the number of classes and the membership of items:
      i. in classification, it is known *a priori* how many classes or groups are present in the data, and which items are members of which class or group;
      ii. in clustering, the number of classes and the membership of items are both unknown;

   (b) in terms of the objective:
      i. in classification, the objective is to classify new items (possibly in the form of a test set) into one of the given classes based upon the experience obtained using a learning set of data;
      ii. clustering falls more into the framework of *exploratory data analysis*, where no prior information is available regarding the class structure of the data;

   (c) in terms of what to classify:
      i. classification deals almost exclusively with classifying *observations*;
      ii. clustering can be applied to clustering *observations* or *variables* or *both observations and variables* simultaneously, depending upon the context.

7. **Overview of Clustering Algorithms:** We will consider the following categories of clustering algorithms:

   (a) Combinatorial clustering algorithms;
   (b) Hierarchical clustering algorithms, including
      i. agglomerative clustering methods, and
      ii. divisive cluster methods;
   (c) Partitioning methods, including
      i. K-means algorithm, and
      ii. K-medoids algorithm;
   (d) Mixture modeling clustering algorithms;
   (e) Density-based clustering algorithms; and
   (f) Spectral clustering algorithms.

II. Dissimilarity and Similarity Measurements and Proximity Matrix

1. **Dissimilarity Measurement:** Many clustering algorithms requires a measurement of the *dissimilarity* or the similarity of one item relative to another item. Let $\mathbf{x}_i, \mathbf{x}_{i'} \in \mathbb{R}^p$.

   (a) *Properties:* Dissimilarity measurement usually satisfy the following properties:

   (i) $d(\mathbf{x}_i, \mathbf{x}_{i'}) \geq 0$;
   (ii) $d(\mathbf{x}_i, \mathbf{x}_i) = 0$;
   (iii) $d(\mathbf{x}_{i'}, \mathbf{x}_i) = d(\mathbf{x}_i, \mathbf{x}_{i'})$; and
   (iv) $d(\mathbf{x}_i, \mathbf{x}_{i'}) \leq d(\mathbf{x}_i, \mathbf{x}_k) + d(\mathbf{x}_k, \mathbf{x}_{i'})$.

   If a dissimilarity measurement d satisfies (i)-(iv), it is said to be a *metric*.
   If d satisfies (i)-(iii) above and does *not* satisfy (iv) above but satisfies

   $$d(\mathbf{x}_i, \mathbf{x}_{i'}) \leq \max\{d(\mathbf{x}_i, \mathbf{x}_k), d(\mathbf{x}_{i'}, \mathbf{x}_k)\},$$

   it is said to be a *ultra-metric*.

   (b) *Examples:* Let $\mathbf{x}_i = (x_{i,1}, \cdots, x_{i,p})^\top \in \mathbb{R}^p$ and $\mathbf{x}_{i'} = (x_{i',1}, \cdots, x_{i',p})^\top \in \mathbb{R}^p$.

   i. Euclidean distance:

   $$d(\mathbf{x}_i, \mathbf{x}_{i'}) = \sqrt{(\mathbf{x}_i - \mathbf{x}_{i'})^\top (\mathbf{x}_i - \mathbf{x}_{i'})} = \sqrt{\sum_{j=1}^{p} (x_{i,j} - x_{i',j})^2};$$

   ii. Mahalanobis distance:

   $$d(\mathbf{x}_i, \mathbf{x}_{i'}) = \sqrt{(\mathbf{x}_i - \mathbf{x}_{i'})^\top \mathbf{S}^{-1} (\mathbf{x}_i - \mathbf{x}_{i'})},$$

   where $\mathbf{S} \in \mathbb{R}^{p \times p}$ is a positive definite matrix;

   iii. Manhattan distance:

   $$d(\mathbf{x}_i, \mathbf{x}_{i'}) = \sum_{j=1}^{p} |x_{i,j} - x_{i',j}|;$$

   iv. Minkowski distance:

   $$d(\mathbf{x}_i, \mathbf{x}_{i'}) = \left[\sum_{j=1}^{p} |x_{i,j} - x_{i',j}|^m \right]^{\frac{1}{m}};$$

   v. Correlation:

   $$d(\mathbf{x}_i, \mathbf{x}_{i'}) = 1 - \rho_{i,i'} = 1 - \frac{s_{i,i'}}{s_i s_{i'}},$$

where $\rho_{i,i'} \in [-1,1]$ is the correlation between the pair of observations $\mathbf{x}_i$ and $\mathbf{x}_{i'}$, and

$$\begin{aligned} s_{i,i'} &= \sum_{j=1}^{p}(x_{i,j}-\bar{x}_i)(x_{i',j}-\bar{x}_{i'}),\\ s_i &= \sqrt{\sum_{j=1}^{p}(x_{i,j}-\bar{x}_i)^2},\\ s_{i'} &= \sqrt{\sum_{j=1}^{p}(x_{i',j}-\bar{x}_{i'})^2},\\ \bar{x}_i &= \frac{1}{p}\sum_{j=1}^{p}x_{i,j},\\ \bar{x}_{i'} &= \frac{1}{p}\sum_{j=1}^{p}x_{i',j}. \end{aligned}$$

A relatively large absolute value of $\rho_{i,i'}$ suggests $\mathbf{x}_i$ and $\mathbf{x}_{i'}$ are "close" to each other, whereas a small correlation ($\rho_{i,i'} \approx 0$) suggests $\mathbf{x}_i$ and $\mathbf{x}_{i'}$ are "far away" from each other. Thus, $1-\rho_{i,i'}$ is taken as a measure of "dissimilarity" between them.

Remark. From the formula above, notice that $\bar{x}_i$ and $\bar{x}_{i'}$ are obtained by averaging over variables, but not the observations, and the scale here matters.

2. Dealing with Ordinal Variables:

(a) *Ordinal Variable:* The values of a *ordinal variable* are often represented as contiguous integers, and the realizable values are considered to be an ordered set.

(b) *Dissimilarity Measurement of Ordinal Variable:* Dissimilarity measurements for ordinal variables are generally defined by replacing their M original values with

$$\frac{i-\frac{1}{2}}{M}, \qquad \text{for all } i = 1,2,\cdots,M,$$

in the prescribed order of their original values. They are then treated as quantitative variables on this scale.

3. Dealing with Unordered Categorical Variable: With unordered categorical variables, the degree-of-difference between pairs of values must be delineated *explicitly.*

If the variable assumes M distinct values, their dissimilarities can be arranged in a symmetric $M \times M$ matrix, denoted by $\mathbf{L}$, with elements

$$L_{r,r'} = L_{r',r} \geq 0, \qquad \text{and} \qquad L_{r,r} = 0,$$

where $L_{r,r'}$ denotes the (r,r')-th entry of $\mathbf{L}$ for all $r, r' = 1,2,\cdots,M$.

The most common choice is $L_{r,r'} = 1$ for all $r \neq r'$, while unequal losses can be used to emphasize some errors more than others.

4. **Proximity Matrix:** Given n observations, $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n \in \mathbb{R}^p$, the *proximity matrix*, denoted by $\mathbf{D}$, is an $n \times n$ matrix with the (i, i')-th entry being

$$d_{i,i'} := d(\mathbf{x}_i, \mathbf{x}_{i'}).$$

In other words, each entry $d_{i,i'}$ is the pairwise dissimilarities between observations $\mathbf{x}_i$ and $\mathbf{x}_{i'}$.

Remark 1. Note that $\mathbf{D}$ is symmetric, and all diagonal elements are

$$d_{i,i} = d(\mathbf{x}_i, \mathbf{x}_i) = 0, \qquad \text{for all } i = 1, 2, \cdots, n.$$

Remark 2. Computing the proximity matrix is the starting point of many clustering procedure.

III. Combinatorial Algorithms

1. **Main Idea:** Combinatorial algorithms directly assign each observation to a group or cluster without regard to a probability model.

2. **Setup:** Let

   (a) $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n$ be n observations to be clustered, and

   (b) K be the number of clusters.

3. **Goal:** We assign each observation to one and only one cluster. These assignments can be characterized by a many-to-one mapping

$$C : \{1, 2, \cdots, n\} \to \{1, 2, \cdots, K\}$$

that assigns the i-th observation to the k-th cluster.

One seeks the particular encoder C^* that achieves a certain required objective, based on the dissimilarity measurements between every pair of observations.

Remark. The encoder $C(i)$ is explicitly delineated by giving the cluster assignment for each observation i. The cluster assignments are adjusted so as to minimize a "loss" function that characterizes the degree to which the clustering goal is not met.

4. **Problem Formulation:** The loss function we choose is the so-called *within-cluster objective function* defined as

$$W(C) := \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} d(\mathbf{x}_i, \mathbf{x}_{i'}). \tag{1}$$

We minimize W through some combinatorial optimization algorithm.

Note that the criterion W characterizes the extent to which observations assigned to the same cluster tend to be close to one another.

Remark. Note that

$$\begin{aligned} T &:= \frac{1}{2}\sum_{i=1}^{n}\sum_{i'=1}^{n} d(\mathbf{x}_i, \mathbf{x}_{i'}) \\ &= \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\left(\sum_{C(i')=k} d(\mathbf{x}_i, \mathbf{x}_{i'}) + \sum_{C(i')\neq k} d(\mathbf{x}_i, \mathbf{x}_{i'})\right) \\ &= \underbrace{\frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\sum_{C(i')=k} d(\mathbf{x}_i, \mathbf{x}_{i'})}_{=:W(C)} + \underbrace{\frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\sum_{C(i')\neq k} d(\mathbf{x}_i, \mathbf{x}_{i'})}_{=:B(C)}, \end{aligned}$$

where T is the sum of all dissimilarity measurements and is constant, and $B(C)$ is called the *between-clsuter objective function*. Note that $B(C)$ tends to be large when observations assigned to different clusters are far apart.

Since

$$W(C) = T - B(C),$$

minimizing W is equivalent to maximizing B.

5. **Issue with Minimizing W:** One minimizes W or equivalently maximizes B over *all* possible assignments of the n data points to K clusters. Unfortunately, such optimization by complete enumeration is feasible only for *very small* data sets. The number of distinct assignments is

$$S(n, K) = \frac{1}{K!}\sum_{k=1}^{K}(-1)^{K-k}\binom{K}{k}k^n, \tag{2}$$

which grows very rapidly with increasing values of n and/or K.

6. **Iterative Greedy Descent Algorithm:** We can find a (locally) optimal cluster assignment using iterative greedy descent, where we assume n or K is small so that the algorithm is feasible. The complete algorithm is outlined in Algorithm 1.

Algorithm 1 Combinatorial Clustering Algorithm

Require: An initial cluster assignment is specified.

1: At each iterative step, the cluster assignments are changed in such a way that the value of the criterion W is improved from its previous value;
2: When the clustering prescription is unable to provide an improvement, the algorithm terminates with the current assignments as its solution.

Remark 1. Since the assignment of observations to clusters at any iteration is a perturbation of that for the previous iteration, only a very small fraction of all possible assignments (2) are examined.

Remark 2. The algorithm described here converges to local optima which may be highly suboptimal when compared to the global optimum.

IV. Hierarchical Clustering

1. **Overview:** Hierarchical clustering produces hierarchical representations in which the clusters at each level of the hierarchy are created by merging clusters at the next lower level:

   (a) At the lowest level, each cluster contains a single observation; and

   (b) At the highest level there is only one cluster containing all of the data.

2. **Types:** There are two types of hierarchical clustering methods — *agglomerative* and *divisive*.

   - *Agglomerative Clustering Methods:* These methods are also called "bottom-up" methods. They start from each item being its own cluster; then, clusters are successively merged until a single cluster remains;
   - *Divisive Cluster Methods:* These methods are also called "top-down" methods. They start with all items as members of a single cluster; then, that single cluster is split into two separate clusters, and so on for every successive cluster, until each item is its own cluster.

3. **Dendrogram:** The end result of all hierarchical clustering methods is a *dendrogram*, i.e., a hierarchical tree diagram, where the k-cluster solution is obtained by merging some of the clusters from the $(k+1)$-cluster solution.

   We assume dendrograms are formed in a *vertical* way.

   (a) *How to Combine:* The dendrogram uses the "*height*" of the linkage criterion at which items or clusters or both are combined together to form a new larger cluster.

   - Items that are similar to each other are combined at *low* heights;
   - Items that are more dissimilar are combined *higher* up the dendrogram.

   The *difference in heights* defines how close items are to each other.

   (b) *How to Form Partitions:* We can "cut" a dendrogram at an appropriate height to form a partition of data into a specified number of groups.

   Draw a horizontal line on the dendrogram at a given height.

   i. The Number of Clusters: The number, K, of vertical lines cut by that horizontal line identifies a K-cluster solution;

   ii. How Clusters Are Determined: The *intersection* of the horizontal line and one of those K vertical lines represents a cluster;

   iii. Cluster Members: The items located at the end of all branches below that intersection constitute the members of the cluster.

4. **Agglomerative Clustering:**

   (a) *General Idea:* Start from each item being a single cluster, and combine two clusters to form a new larger cluster.

(b) *Linkage Methods:* Three major methods to compute the distance between two clusters are the following:

i. *Single linkage* uses a minimum-distance metric between clusters;

ii. *Complete linkage* uses a maximum-distance metric; and

iii. *Average linkage* computes the average distance between all pairs of items within the two different clusters, one item from each cluster.

There is also a *weighted version* of average linkage, where the weights reflect the (possibly disparate) sizes of the clusters in question.

(c) *Algorithm:* The complete algorithm using three different linkage methods is shown in Algorithm 2.

Algorithm 2 Agglomerative Hierarchical Clustering

Require: Items to be clustered $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n$, where n is the number of observations and is also the number of initial clusters.

1: Compute $\mathbf{D} \in \mathbb{R}^{n \times n}$, the matrix of dissimilarities between the n clusters, where the (i, i')-th entry of $\mathbf{D}$ is $d_{i,i'} := d(\mathbf{x}_i, \mathbf{x}_{i'})$, for all $i, i' = 1, 2, \cdots, n$.

2: Find the *smallest* dissimilarity, say, $d_{I,J}$, in $\mathbf{D} = \mathbf{D}^{(1)}$. Merge clusters I and J to form a new cluster (I, J).

3: Compute dissimilarities, $d_{(I,J),K}$, between the new cluster (I, J) and all other clusters $K \neq I, J$. These dissimilarities depend upon which linkage method is used. For all clusters $K \neq I, J$, we have the following linkage options:

i. Single linkage: $d_{(I,J),K} = \min\{d_{I,K}, d_{J,K}\}$;

ii. Complete linkage: $d_{(I,J),K} = \max\{d_{I,K}, d_{J,K}\}$;

iii. Average linkage: $d_{(I,J),K} = \frac{1}{n_{(I,J)} n_K} \sum_{i \in I, j} \sum_{k \in K} d_{i,k}$, where $n_{(I,J)}$ and n_K are the numbers of items in clusters (I, J) and K, respectively.

4: Form a new $((n-1) \times (n-1))$-matrix, $\mathbf{D}^{(2)}$, by deleting rows and columns I and J and adding a new row and column (I, J) with dissimilarities computed from Step 3.

5: Repeat Steps 2, 3, and 4, a total of $n - 1$ times. At the i-th step, $\mathbf{D}^{(i)}$ is a symmetric $((n-i+1) \times (n-i+1))$-matrix, for all $i = 1, 2, \cdots, n$. At the last step where $i = n$, $\mathbf{D}^{(n)} = 0$, and all items are merged together into a single cluster.

6: **return** The following items are returned:

i. list of which clusters are merged at each step,

ii. the value (or height) of the dissimilarity of each merge, and

iii. a dendrogram to summarize the clustering procedure.

(d) *Properties:*

i. No one of the linkage methods described above is uniformly best for all clustering problems;
ii. If the data dissimilarities $\{d(\mathbf{x}_i, \mathbf{x}_{i'})\}_{i,i'=1,2,\cdots,n}$ exhibit a strong clustering tendency, with each of the clusters being compact and well separated from others, then all three linkage methods produce similar results;
iii. The dendrograms from single-linkage and complete-linkage methods are *invariant* under monotone transformations of the pairwise dissimilarities, but this property does *not* hold for the average-linkage method;
iv. Single-linkage often leads to long "chains" of clusters, joined by singleton points near each other;
v. Complete-linkage tends to produce many small, compact clusters;
vi. Average linkage is dependent upon the *size* of the clusters, whereas single and complete linkage, which depend only upon the smallest or largest dissimilarity, respectively, are *not*.

5. **Cluster Diameter:** Let $A \subset \{1, 2, \cdots, n\}$ and $\{\mathbf{x}_i\}_{i\in A}$ denote a cluster of observations. Its *cluster diameter* is defined as

$$D_A := \max_{i\in A, i'\in A} d(\mathbf{x}_i, \mathbf{x}_{i'}). \tag{3}$$

Remark. Single linkage can produce clusters with very large diameters, whereas complete linkage can produce clusters with small diameters.

6. **Divisive Clustering:** *Divisive clustering* is a "top-down" method for hierarchical clustering.

At each step, the items are divided into a "splinter" group (say, Cluster A) and the "remainder" (say, Cluster B). The complete algorithm is shown in Algorithm 3.

Algorithm 3 Divisive Clustering

1: The splinter group is initiated by extracting the item that has the largest average dissimilarity from all other items in the data set; that item is set up as Cluster A;
2: Given this separation of the data into Clusters A and B, we next compute, for each item in Cluster B, the following two quantities:

   (1) the average dissimilarity between that item and all other items in Cluster B, and

   (2) the average dissimilarity between that item and all items in cluster A.

3: Compute the difference between (1) and (2) for each item in Cluster B:

   (a) If all differences are negative, we stop the algorithm;

   (b) If any of these differences are positive (indicating that the item in B is closer on average to Cluster A than to the other items in Cluster B), we take the item in B with the largest positive difference, move it to A, and repeat the procedure.

4: Steps 2 and 3 can then be used to obtain binary splits of each of the clusters A and B separately, until each observation becomes a single cluster.

Remark. This algorithm provides a binary split of the data into two clusters.

V. Partitioning Methods

1. **Overview:** *Partitioning methods* split the data items into a pre-determined number K of clusters. Given K, we seek to partition the data into K clusters so that

   (a) the items within each cluster are similar to each other, whereas

   (b) the items from different clusters are quite dissimilar.

   Remark. In contrast to hierarchical methods, in partitioning methods, there is no hierarchical relationship between the K-cluster solution and the $(K+1)$-cluster solution; that is, the K-cluster solution is *not* the initial step for the $(K+1)$-cluster solution.

2. **K-Means Clustering:**

   (a) *Goal:* To find clusters that minimize within-cluster sum of squares:

$$\begin{aligned} &\underset{C_1,C_2,\cdots,C_K}{\text{minimize}} \sum_{k=1}^{K} \sum_{i\in C_k} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|_2^2, \\ &\text{subject to } C_1, C_2, \cdots, C_K \text{ is a partition of } \{1, 2, \cdots, n\}, \\ &\qquad \boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_K \in \mathbb{R}^p. \end{aligned} \tag{4}$$

(b) *Necessity to Use a Greedy Algorithm:* The problem (4) is a combinatorial problem. Finding a global minimum requires a computationally intractable combinatorial search. Most K-means algorithms use a *greedy iterative heuristic* that examine only a small number of possible clusters.

(c) *Equivalent Problem:* The problem (4) is equivalent to

$$\begin{aligned} &\text{minimize} \sum_{k=1}^{K} \sum_{i \in C_k} \left\| \mathbf{x}_i - \frac{1}{|C_k|} \sum_{i' \in C_k} \mathbf{x}_{i'} \right\|_2^2, \\ &\text{subject to } C_1, C_2, \cdots, C_K \text{ is a partition of } \{1, 2, \cdots, n\}. \end{aligned} \tag{5}$$

Thus, the main difficulty stems from determining $C_1, C_2, \cdots, C_K$ with respect to $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \cdots, \boldsymbol{\mu}_K$.

(d) *Lloyd's Algorithm:* Lloyd's algorithm, provided in Algorithm 4, is a greedy algorithm to solve the problem (5).

Algorithm 4 Lloyd's Algorithm

Require: Number of clusters, K;
Require: Data to be clustered, $\{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$;
Require: Initial cluster assignment.
1: **repeat**
2: Update

$$\widehat{\boldsymbol{\mu}}_k \longleftarrow \frac{1}{|\widehat{C}_k|} \sum_{i \in \widehat{C}_k} \mathbf{x}_i, \qquad \text{for all } k = 1, \cdots, K;$$

3: Initialize $\widehat{C}_k = \emptyset$ for all $k = 1, \cdots, K$;
4: **for** $i = 1, \cdots, n$ **do**
5: Assign

$$\hat{k} \longleftarrow \underset{k \in \{1, \cdots, K\}}{\arg\min} \|\mathbf{x}_i - \widehat{\boldsymbol{\mu}}_k\|_2^2;$$

6: Update $\widehat{C}_{\hat{k}} \leftarrow \widehat{C}_{\hat{k}} \cup \{i\}$.
7: **end for**
8: **until** Convergence is reached
9: **return** Cluster assignments $\widehat{C}_1, \cdots, \widehat{C}_K$ and cluster centers $\widehat{\boldsymbol{\mu}}_1, \cdots, \widehat{\boldsymbol{\mu}}_K$.

(e) *Termination Criterion:* One can choose the following convergence criteria:

i. Class assignments are unchanged; or
ii. The sum-of-squares objective function does *not* change by more than a pre-specified tolerance parameter;

(f) *Convergence Property of Lloyd's Algorithm:*

i. Since the sum-of-squares objective function is non-increasing in each iteration, and is bounded from below. Thus, the algorithm must converge.

ii. Clustering will stabilize after at most 2^n iterations, which means that the algorithm terminates in a finite number of iterations.

(g) *Non-uniqueness of the Solution:* The solution to (5), a configuration of items into K clusters, will typically *not* be unique; the algorithm will only find a *local minimum* of the objective function in (5).

Remark. Because of the non-uniquness of the solution, the algorithm should be run using different initial random assignments of the items to K clusters (or by randomly selecting K initial centroids) in order to find the lowest minimum of the objective function.

(h) *Cyclic Block Coordinate Descent View of Lloyd's Algorithm:* Lloyd's algorithm can be viewed as a cyclic block coordinate descent. Write the objective function in (4) as

$$f(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \cdots, \boldsymbol{\mu}_K, C_1, C_2, \cdots, C_K).$$

The algorithm alternates

$$(\widehat{\boldsymbol{\mu}}_1, \widehat{\boldsymbol{\mu}}_2, \cdots, \widehat{\boldsymbol{\mu}}_K) = \underset{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \cdots, \boldsymbol{\mu}_K}{\arg\min} f(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \cdots, \boldsymbol{\mu}_K, \widehat{C}_1, \widehat{C}_2, \cdots, \widehat{C}_K),$$
$$(\widehat{C}_1, \widehat{C}_2, \cdots, \widehat{C}_K) = \underset{C_1, C_2, \cdots, C_K}{\arg\min} f(\widehat{\boldsymbol{\mu}}_1, \widehat{\boldsymbol{\mu}}_2, \cdots, \widehat{\boldsymbol{\mu}}_K, C_1, C_2, \cdots, C_K).$$

(i) *Drawbacks:* K-means clustering algorithm has the following potential drawbacks:

i. K-means algorithm is only appropriate when the dissimilarity measurement is taken to be squared Euclidean distance, which requires that all of the variables have to be of the quantitative type.

ii. Using squared Euclidean distance places the highest influence on the largest distances, which causes the procedure to lack robustness against outliers that produce very large distances.

3. K-Medoids Algorithm:

(a) *Motivation:* In the K-means clustering algorithm,

i. only the minimization step of (4) involves the squared Euclidean distance, and

ii. due to the choice of the squared Euclidean distance, the cluster representation $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \cdots, \boldsymbol{\mu}_K$ are taken to be the means of the currently assigned clusters.

The K-medoids algorithm generalizes the K-means clustering algorithm above, and

i. replaces the squared Euclidean distance by the dissimilarity measurement, and

ii. restricts the center of each cluster to be one of the observations assigned to the cluster.

(b) *Medoid:* The *medoid* of a cluster is the observation within this cluster that *minimizes* the total dissimilarity to all other observations within that cluster.

(c) *Algorithm:*

Algorithm 5 K-Medoids Algorithm

Require: Number of clusters, K;

Require: Proximity matrix, $\mathbf{D} \in \mathbb{R}^{n\times n}$ with the (i,i')-th entry being $d_{i,i'} := d(\mathbf{x}_i, \mathbf{x}_{i'})$ for all $i, i' = 1, 2, \cdots, n$.

1: Form an initial assignment of the items into K clusters;

2: Locate the medoid for each of the K clusters. That is, we solve the following optimization problem

$$i_k^* = \underset{\{i \,|\, i\in C_k\}}{\arg\min} \sum_{\{i' \,|\, i'\in C_k\}} d(\mathbf{x}_i, \mathbf{x}_{i'}), \qquad \text{for all } k = 1, 2, \cdots, K,$$

where $C_1, C_2, \cdots, C_K$ form a partition of $\{1, 2, \cdots, n\}$. Then, $\boldsymbol{\mu}_k = \mathbf{x}_{i_k^*}$, for all $k = 1, 2, \cdots, K$, are the current estimates of the cluster centers;

3: Given a current set of cluster centers $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \cdots, \boldsymbol{\mu}_K\}$, assign each observation to the closest (current) cluster center, i.e.,

$$\underset{k\in\{1,2,\cdots,K\}}{\arg\min} \; d(\mathbf{x}_i, \boldsymbol{\mu}_k);$$

4: Repeat the preceding two steps until no further reassignment of items takes place.

(d) *Equivalent Optimization Problem:* Alternating between 2 and 3 in Algorithm 5 represents a particular heuristic search strategy for solving

$$\underset{C_1, C_2, \cdots, C_K, i_1, i_2, \cdots, i_K}{\text{minimize}} \sum_{k=1}^{K} \sum_{\{i \,|\, i\in C_k\}} d(\mathbf{x}_i, \mathbf{x}_{i_k}).$$

(e) *Comparison to K-Means Algorithm:*

i. K-medoids algorithm searches for K medoids among the items in the data, but the K-means algorithm searches for K centroids;

ii. K-medoids algorithm uses a dissimilarity-based distance, but the K-means uses the squared Euclidean distance;

iii. K-medoids algorithm is more robust to data anomalies such as outliers and missing values;

iv. K-medoids algorithm is computationally more intensive than K-means algorithm.

4. **Partitioning Around Medoids (PAM) Algorithm:** PAM algorithm is a modification of the K-medoids algorithm by introducing a swapping strategy: *replace the medoid of each cluster by another item in that cluster only if such a swap can reduce the value of the objective function.*

Algorithm 6 Partitioning Around Medoid Algorithm

Require: Number of clusters, K;

Require: Proximity matrix, $\mathbf{D} \in \mathbb{R}^{n\times n}$ with the (i,i')-th entry being $d_{i,i'} := d(\mathbf{x}_i, \mathbf{x}_{i'})$ for all $i, i' = 1, 2, \cdots, n$.

1: Form an initial assignment of the items into K clusters;

2: Locate the medoid for each of the K clusters. That is, we solve the following optimization problem

$$i_k^* = \underset{\{i \,|\, \mathbf{x}_i \in C_k\}}{\arg\min} \sum_{\{i' \,|\, \mathbf{x}_{i'} \in C_k\}} d(\mathbf{x}_i, \mathbf{x}_{i'}), \qquad \text{for all } k = 1, 2, \cdots, K,$$

where $C_1, C_2, \cdots, C_K$ form a partition of $\{1, 2, \cdots, n\}$. Then, $\boldsymbol{\mu}_k = \mathbf{x}_{i_k^*}$, for all $k = 1, 2, \cdots, K$, are the current estimates of the cluster centers;

3: For each cluster, swap the medoid with the non-medoid item that gives the largest reduction in

$$\text{ESS}_{\text{medoid}} := \sum_{k=1}^{K} \sum_{\{i \,|\, i \in C_k\}} d(\mathbf{x}_i, \mathbf{x}_{i_k});$$

4: Repeat the preceding two steps until no further reduction in $\text{ESS}_{\text{medoid}}$ takes place.

Remark. Similar to the K-medoids algorithm, PAM algorithm is computationally intensive and performs well on small datasets but are *not* efficient on large datasets.

5. **How to Choose the Value of K:** A choice for the number of clusters K depends on the goal and is usually defined as part of the problem. There are several ways of determining the value of K.

   (a) *Known K:* In some scenarios, the value of K is pre-determined based on the problem at hand.

   (b) *Elbow Point Method:* We plot the within-cluster dissimilarity W_K as a function of the number of clusters K, for $K \in \{1, 2, \cdots, K_{\max}\}$. Typically, we have

   $$W_1 \geq W_2 \geq \cdots \geq W_{K_{\max}}.$$

   If there are actually K^* distinct groupings of the observations, the plot of W_K against K should exhibit a shape decrease at K^*. A natural estimate of K^* is then obtained by identifying a elbow point in this plot.

   (c) *Average Silhouette Width:* See below.

6. **Silhouette Plot and Coefficient:**

(a) *Notation:* Let

i. $\mathcal{C}_K$ be a particular clustering of the data into K clusters;

ii. $C(i)$ denote the cluster containing the i-th observation;

iii. a_i be the average dissimilarity of that i-th item to all other members of the same cluster $C(i)$;

iv. $d(\mathbf{x}_i, \mathbf{x}_C)$ be the average dissimilarity of the i-th observation to all members of C, where C may be some cluster other than $C(i)$;

v. $b_i := \min_{C \neq C(i)} d(\mathbf{x}_i, \mathbf{x}_C)$ be the minimal average dissimilarity of the i-th observation to all other clusters other than $C(i)$.

Remark. If $b_i = d(\mathbf{x}_i, \mathbf{x}_{C^*})$, then, Cluster C^* is called the *neighbor* of the i-th observation and is regarded as the second-best cluster for the i-th observation.

(b) *i-th Silhouette Value:* The *i-th silhouette value* (or *width*) is given by

$$s_i(\mathcal{C}_K) := s_{i,K} = \frac{b_i - a_i}{\max\{a_i, b_i\}}. \tag{6}$$

Note that $s_{i,K} \in [-1, 1]$.

(c) *Properties of $s_{i,K}$:*

i. Large positive values of $s_{i,K}$, indicate $a_i \approx 0$ and that the i-th observation is well-clustered;

ii. Large negative values of $s_{i,K}$ indicate $b_i \approx 0$ and poor clustering, and

iii. $s_{i,K} \approx 0$, i.e., $a_i \approx b_i$, indicates that the i-th item lies between two clusters.

Remark 1. The observations corresponding to negative silhouette values are considered to be borderline allocations, and are neither well-clustered nor assigned by the clustering process to an alternative cluster.

Remark 2. If $\max_i\{s_{i,K}\} < 0.25$, this indicates either that there are *no* definable clusters in the data or that, even if there are, the clustering procedure has *not* found it.

(d) *Silhouette Plot:* A *silhouette plot* is a bar plot of all the $\{s_{i,K}\}_{i=1}^n$ values after they are ranked in decreasing order within each cluster, where the length of the i-th bar is $s_{i,K}$.

(e) *Average Silhouette Width:* The *average silhouette width*, denoted by $\bar{s}_K$, is the average of all the $\{s_{i,K}\}_{i=1}^n$ values.

Remark. The statistic $\bar{s}_K$ has been found to be a very useful indicator of the merit of the clustering $\mathcal{C}_K$. The average silhouette width has also been used to choose the value of K by finding K to maximize $\bar{s}_K$.

(f) *Silhouette Coefficient:* Define the *silhouette coefficient* to be

$$\text{SC} := \max_K\{\bar{s}_K\}.$$

A subjective interpretation of SC is provided in Table 1.

SC value	Interpretation
> 0.70	A strong structure has been found
$(0.50, 0.70]$	A reasonable structure has been found
$(0.25, 0.50]$	The structure is weak and could be artificial
≤ 0.25	No substantial structure has been found

Table 1: Interpretation of silhouette coefficient.

IV. Clustering Based on Mixture Models

1. **Setup and Motivation:** Suppose $X \overset{\text{i.i.d}}{\sim} p(\cdot \,|\, \boldsymbol{\theta})$, where $p(\cdot \,|\, \boldsymbol{\theta})$ is a density function with the unknown parameter $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, and $\boldsymbol{\Theta}$ is the parameter space.

   Assume there is no missing values in X. The *complete-date likelihood function* is

   $$L(\boldsymbol{\theta} \,|\, X) := p(X \,|\, \boldsymbol{\theta}).$$

   Now, suppose some components of X are missing and write

   $$X = (X_{\text{obs}}^\top, X_{\text{mis}}^\top)^\top,$$

   where X_{obs} is the observed part of X, and X_{mis} is the missing part of X. The likelihood function for the observed data is

   $$L_{\text{obs}}(\boldsymbol{\theta} \,|\, X_{\text{obs}}) := \int p(X_{\text{obs}}, x_{\text{mis}} \,|\, \boldsymbol{\theta}) \, \mathrm{d}x_{\text{mis}}.$$

   Estimating $\boldsymbol{\theta}$ via maximizing $L_{\text{obs}}(\cdot \,|\, X_{\text{obs}})$ directly is usually hard.

2. **EM Algorithm:** The *EM algorithm* is a two-step iterative process, incorporating

   (a) an *expectation step* (E-step): compute the conditional expectation of the complete-data log-likelihood given the observed data and the current parameter estimate, and then

   (b) a *maximization step* (M-step): update the parameter estimate by maximizing the conditional expectation from the E-step.

3. **Derivation of EM Algorithm:** Since

   $$p(X \,|\, \boldsymbol{\theta}) = p(X_{\text{obs}}, X_{\text{mis}} \,|\, \boldsymbol{\theta}) = p(X_{\text{mis}} \,|\, X_{\text{obs}}, \boldsymbol{\theta}) \times p(X_{\text{obs}} \,|\, \boldsymbol{\theta}),$$

   or equivalently,

   $$p(X_{\text{obs}} \,|\, \boldsymbol{\theta}) = \frac{p(X_{\text{obs}}, X_{\text{mis}} \,|\, \boldsymbol{\theta})}{p(X_{\text{mis}} \,|\, X_{\text{obs}}, \boldsymbol{\theta})},$$

the log-likelihood function for the observed data is

$$\begin{aligned}\ell(\boldsymbol{\theta} \mid X_{\text{obs}}) &= \log p(X_{\text{obs}}, X_{\text{mis}} \mid \boldsymbol{\theta}) - \log p(X_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}) \\ &= \ell(\boldsymbol{\theta} \mid X) - \log p(X_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}),\end{aligned} \tag{7}$$

where $\ell(\cdot \mid X)$ is the log-likelihood function for the complete data, which may be easy to compute, and $\log p(X_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta})$ is the part of the log-likelihood function for the complete data due to the missing data.

Take the expectation of both sides of (7) with respect to the conditional distribution of the missing variable conditioning on the observed part and the current parameter estimate $\boldsymbol{\theta}'$, we have

$$\begin{aligned}\ell(\boldsymbol{\theta} \mid X_{\text{obs}}) &= \mathbb{E}[\ell(\boldsymbol{\theta} \mid X) \mid X_{\text{obs}}, \boldsymbol{\theta}'] - \mathbb{E}[\log p(X_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}) \mid X_{\text{obs}}, \boldsymbol{\theta}'] \\ &= Q(\boldsymbol{\theta}, \boldsymbol{\theta}') - R(\boldsymbol{\theta}, \boldsymbol{\theta}'),\end{aligned}$$

where

$$\begin{aligned}Q(\boldsymbol{\theta}, \boldsymbol{\theta}') &:= \mathbb{E}[\ell(\boldsymbol{\theta} \mid X) \mid X_{\text{obs}}, \boldsymbol{\theta}'] \\ &= \int \ell(\boldsymbol{\theta} \mid X) p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}') \mathrm{d}x_{\text{mis}}\end{aligned}$$

and

$$\begin{aligned}R(\boldsymbol{\theta}, \boldsymbol{\theta}') &:= \mathbb{E}[\log p(X_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}) \mid X_{\text{obs}}, \boldsymbol{\theta}'] \\ &= \int p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}') \log p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}) \mathrm{d}x_{\text{mis}}.\end{aligned}$$

Now, for $R(\boldsymbol{\theta}, \boldsymbol{\theta}')$, we notice

$$\begin{aligned}R(\boldsymbol{\theta}, \boldsymbol{\theta}') &= \int p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}') \log p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}) \mathrm{d}x_{\text{mis}} \\ &= \int p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}') \log \left(p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}) \frac{p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}')}{p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}')} \right) \mathrm{d}x_{\text{mis}} \\ &= \int p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}') \log \left(\frac{p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta})}{p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}')} \right) \mathrm{d}x_{\text{mis}} \\ &\qquad + \int p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}') \log p(x_{\text{mis}} \mid X_{\text{obs}}, \boldsymbol{\theta}') \mathrm{d}x_{\text{mis}} \\ &= -\operatorname{KL}\big(p(\cdot \mid X_{\text{obs}}, \boldsymbol{\theta}') \,\|\, p(\cdot \mid X_{\text{obs}}, \boldsymbol{\theta})\big) + R(\boldsymbol{\theta}', \boldsymbol{\theta}') \\ &\leq R(\boldsymbol{\theta}', \boldsymbol{\theta}'),\end{aligned}$$

where $\operatorname{KL}(p \,\|\, q) := \int p(x) \log(p(x)/q(x)) \mathrm{d}x$ is the Kullback-Leibler divergence between density functions p and q, and $\operatorname{KL}(p \,\|\, q) \geq 0$ always holds due to Jensen's inequality.

Given the current value $\boldsymbol{\theta}'$, if we can find $\boldsymbol{\theta}'' \in \boldsymbol{\Theta}$ such that $Q(\boldsymbol{\theta}'', \boldsymbol{\theta}') > Q(\boldsymbol{\theta}', \boldsymbol{\theta}')$, we must have

$$\begin{aligned}\ell(\boldsymbol{\theta}'' \mid X_{\text{obs}}) - \ell(\boldsymbol{\theta}' \mid X_{\text{obs}}) &= \big[Q(\boldsymbol{\theta}'', \boldsymbol{\theta}') - R(\boldsymbol{\theta}'', \boldsymbol{\theta}')\big] - \big[Q(\boldsymbol{\theta}', \boldsymbol{\theta}') - R(\boldsymbol{\theta}', \boldsymbol{\theta}')\big] \\ &= \big[Q(\boldsymbol{\theta}'', \boldsymbol{\theta}') - Q(\boldsymbol{\theta}', \boldsymbol{\theta}')\big] - \big[R(\boldsymbol{\theta}'', \boldsymbol{\theta}') - R(\boldsymbol{\theta}', \boldsymbol{\theta}')\big] \\ &> 0,\end{aligned}$$

that is, we are keeping increasing the value of the log-likelihood function for the observed data.

The complete EM algorithm is given in Algorithm 7.

Algorithm 7 EM Algorithm

Require: Initial guess of the parameter value, $\boldsymbol{\theta}^{(0)}$;

1: For $m = 0, 1, 2, \cdots$, iterate between the following two steps:

(a) *E-step:* Compute

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(m)}) = \mathbb{E}[\ell(\boldsymbol{\theta} \mid X) \mid X_{\text{obs}}, \boldsymbol{\theta}^{(m)}];$$

(b) *M-step:* Compute

$$\boldsymbol{\theta}^{(m+1)} := \arg\max_{\boldsymbol{\Theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(m)});$$

2: Stop when convergence of the log-likelihood is attained.

Remark. In the M-step above, we do *not* need to solve the maximization exactly, but just need an estimate $\boldsymbol{\theta}^{(m+1)}$ satisfying $Q(\boldsymbol{\theta}^{(m+1)}, \boldsymbol{\theta}^{(m)}) > Q(\boldsymbol{\theta}^{(m)}, \boldsymbol{\theta}^{(m)})$.

4. **Convergence Property:** Under certain mild regularity conditions, the EM algorithm is guaranteed to converge to a local maximum of the log-likelihood function.

   Remark. Local convergence of the log-likelihood function does *not* imply local convergence of the parameter estimates, although the latter can be achieved under additional regularity conditions.

5. **Comments:**

   (a) The EM algorithm possesses reliable convergence properties and low cost per iteration, does *not* require much storage space, and is easy to program;

   (b) The EM algorithm may be extremely slow to converge;

   (c) Because convergence is guaranteed only to a *local* maximum, and because likelihood surfaces often possess many local maxima, it is usually necessary to run the EM algorithm using different starting points to try to find a global maximum of the log-likelihood function.

6. **Application of EM Algorithm in Fitting Finite Mixture Models:**

   (a) *Model:* Suppose a density p is a mixture of K component density functions, $p_1, p_2, \cdots, p_K$, that is,

$$p(\mathbf{x} \mid \boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k p_k(\mathbf{x} \mid \boldsymbol{\theta}_k), \qquad \text{for all } \mathbf{x},$$

where $\pi_k \geq 0$ for all $k = 1, 2, \cdots, K$ and $\sum_{k=1}^{K} \pi_k = 1$, $\boldsymbol{\theta}_k$ denotes the parameters associated with the k-th component density function, and $\boldsymbol{\theta} := (\{\pi_k\}_{k=1}^K, \{\boldsymbol{\theta}_k\}_{k=1}^K)$ denotes the set of all parameter in this mixture model.

(b) *Observed, Missing and Complete Data:* Let $\mathbf{x}_{1,\text{obs}}, \mathbf{x}_{2,\text{obs}}, \cdots, \mathbf{x}_{n,\text{obs}}$ be n i.i.d observed samples, and $\mathbf{x}_{i,\text{mis}}$ be the missing component associated with $\mathbf{x}_{i,\text{obs}}$, for all $i = 1, 2, \cdots, n$. Here,

$$\mathbf{x}_{i,\text{mis}} := (x_{i,1,\text{mis}}, x_{i,2,\text{mis}}, \cdots, x_{i,K,\text{mis}})^\top \in \{0,1\}^K$$

and

$$x_{i,k,\text{mis}} = \begin{cases} 1, & \text{if } \mathbf{x}_{i,\text{obs}} \text{ belongs to the } k\text{-component}, \\ 0, & \text{otherwise}, \end{cases}$$

for all $i = 1, 2, \cdots, n$ and $k = 1, 2, \cdots, K$. As a consequence, the complete data for the i-th observation is

$$\mathbf{x}_i := (\mathbf{x}_{i,\text{obs}}^\top, \mathbf{x}_{i,\text{mis}}^\top)^\top,$$

for all $i = 1, 2, \cdots, n$.

(c) *Assumptions on Missing Data:* Assume that $\mathbf{x}_{i,\text{mis}}$ is a realization of a K-class multinomial distribution with cell probabilities given by $(\pi_1, \pi_2, \cdots, \pi_K)$.

(d) *Log-likelihood Function for Complete Data:* The log-likelihood function for complete data is given by

$$\ell(\boldsymbol{\theta} \,|\, \mathbf{x}_1, \cdots, \mathbf{x}_n) := \sum_{i=1}^{n} \sum_{k=1}^{K} x_{i,k,\text{mis}} \log\big(\pi_k p_k(\mathbf{x}_{i,\text{obs}} \,|\, \boldsymbol{\theta}_k)\big). \tag{8}$$

(e) *Derivation of E-step:* Given the current value of the parameter $\boldsymbol{\theta}'$, we compute

$$\begin{aligned} Q(\boldsymbol{\theta}, \boldsymbol{\theta}') &= \mathbb{E}[\ell(\boldsymbol{\theta} \,|\, \mathbf{x}_1, \cdots, \mathbf{x}_n) \,|\, \mathbf{x}_{1,\text{obs}}, \mathbf{x}_{2,\text{obs}}, \cdots, \mathbf{x}_{n,\text{obs}}, \boldsymbol{\theta}'] \\ &= \sum_{i=1}^{n} \sum_{k=1}^{K} \mathbb{E}[x_{i,k,\text{mis}} \,|\, \mathbf{x}_{1,\text{obs}}, \mathbf{x}_{2,\text{obs}}, \cdots, \mathbf{x}_{n,\text{obs}}, \boldsymbol{\theta}'] \log\big(\pi_k p_k(\mathbf{x}_{i,\text{obs}} \,|\, \boldsymbol{\theta}_k)\big) \\ &= \sum_{i=1}^{n} \sum_{k=1}^{K} \hat{x}_{i,k,\text{mis}} \log\big(\pi_k p_k(\mathbf{x}_{i,\text{obs}} \,|\, \boldsymbol{\theta}_k)\big), \end{aligned}$$

where

$$\hat{x}_{i,k,\text{mis}} := \frac{\pi_k' p_k(\mathbf{x}_{i,\text{obs}} \,|\, \boldsymbol{\theta}_k')}{\sum_{\ell=1}^{K} \pi_\ell' p_\ell(\mathbf{x}_{i,\text{obs}} \,|\, \boldsymbol{\theta}_\ell')},$$

and π_k''s and $\boldsymbol{\theta}_k'$'s are the corresponding values in $\boldsymbol{\theta}'$.

(f) *Derivation of M-step:* The M-step then takes the probabilities $\{\hat{x}_{i,k,\text{mis}}\}$ provided by the E-step and updates the parameter values by maximizing (8) with respect to $\{\pi_k\}_{k=1}^K$ and $\{\boldsymbol{\theta}_k\}_{k=1}^K$.
In particular, the M-step outcome for the mixture proportions $\{\pi_k\}_{k=1}^K$ is given by

$$\pi_k'' = \frac{1}{n}\sum_{i=1}^n \hat{x}_{i,k,\text{mis}}, \qquad \text{for all } k = 1, 2, \cdots, K.$$

Remark. The maximum likelihood determination of the density component for the i-th observation is the component corresponding to the largest value of $\hat{x}_{i,k,\text{mis}}$.

7. **Connection between EM Algorithm and K-Means Algorithm:** K-means algorithm can be viewed as the limit of the EM algorithm for the Gaussian finite mixture model where the component covariances are assumed to be all equal to $\sigma^2\mathbf{I}_p$ as $\sigma^2 \to 0$, where we assume $\mathbf{x}_i \in \mathbb{R}^p$ for all $i = 1, 2, \cdots, n$.

Let the k-th component density function be

$$p_k(\mathbf{x} \mid \boldsymbol{\mu}_k, \sigma^2) = \frac{1}{(\sqrt{2\pi\sigma^2})^p}\exp\left(-\frac{1}{2\sigma^2}(\mathbf{x}-\boldsymbol{\mu}_k)^\top(\mathbf{x}-\boldsymbol{\mu}_k)\right), \qquad \text{for all } \mathbf{x} \in \mathbb{R}^p,$$

where $\sigma > 0$ is known.

Then, using the notation above, given the value of $\boldsymbol{\theta}$ at the m-th iteration, the E-step is to update

$$\begin{aligned}\hat{x}_{i,k,\text{mis}}^{(m+1)} &= \frac{\pi_k^{(m)} p_k(\mathbf{x}_{i,\text{obs}} \mid \boldsymbol{\theta}_k^{(m)})}{\sum_{\ell=1}^K \pi_\ell^{(m)} p_\ell(\mathbf{x}_{i,\text{obs}} \mid \boldsymbol{\theta}_\ell^{(m)})} \\ &= \frac{\pi_k^{(m)} \exp\big(-\frac{1}{2\sigma^2}(\mathbf{x}-\boldsymbol{\mu}_k^{(m)})^\top(\mathbf{x}-\boldsymbol{\mu}_k^{(m)})\big)}{\sum_{\ell=1}^K \pi_\ell^{(m)} \exp\big(-\frac{1}{2\sigma^2}(\mathbf{x}-\boldsymbol{\mu}_k^{(m)})^\top(\mathbf{x}-\boldsymbol{\mu}_k^{(m)})\big)}.\end{aligned}$$

Now, letting $\sigma^2 \to 0^+$, we have

$$\hat{x}_{i,k,\text{mis}}^{(m+1)} \to \begin{cases} 1, & \text{if } k = \arg\min_{k=1,2,\cdots,K} \|\mathbf{x}_i - \boldsymbol{\mu}_k^{(m)}\|^2 \\ 0, & \text{otherwise.}\end{cases},$$

Thus, if we consider a restricted version of the Gaussian mixture model with the covariance matrix $\sigma^2\mathbf{I}_p$ known, then the steps of the EM algorithm become approximately the same as those of K-means algorithm as $\sigma^2 \to 0^+$.

Remark. If we simply compare the two algorithms, we can also see that the EM algorithm is like a "soft" version of the K-means algorithm:

(a) K-means assigns each observation to exactly one cluster, whereas

(b) Gaussian EM gives a conditional probability distribution over K clusters.

V. Density-based Spatial Clustering of Applications with Noise (DBSCAN)

1. **Main Idea:** DBSCAN is a non-parametric density-based clustering algorithm. Given a set of points in some space, it groups together points that are closely packed together, marking points that lie alone in low-density regions as outliers.

2. **Basic Assumption:** The assumption underlying DBSCAN is that the clusters are dense regions in space separated by regions of lower density.

3. **Setup:** Let

   (a) $\mathcal{D} := \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$ be a set of data points to be clustered,

   (b) $\varepsilon > 0$, and

   (c) $n_{\min}$ be a strictly positive integer.

4. **ε-neighborhood of a Point:** Let $\mathbf{x}$ be a point in space. The ε-neighborhood of $\mathbf{x}$, denoted by $\mathcal{N}_\varepsilon(\mathbf{x})$, is defined as

$$\mathcal{N}_\varepsilon(\mathbf{x}) := \Big\{ \mathbf{x}_i \in \mathcal{D} \,\Big|\, d(\mathbf{x}, \mathbf{x}_i) \leq \varepsilon \Big\}. \tag{9}$$

5. **Directly Density-reachable Points:** A point $\mathbf{x}$ is *directly density-reachable* from a point $\mathbf{y}$ with respect to ε and $n_{\min}$ if

$$\mathbf{x} \in \mathcal{N}_\varepsilon(\mathbf{y}), \qquad \text{and} \qquad |\mathcal{N}_\varepsilon(\mathbf{y})| \geq n_{\min}.$$

   Remark 1. Suppose $\mathbf{x}$ and $\mathbf{y}$ are both *core points*, meaning that they are points inside of the cluster, direct-reachability is symmetric for them.

   If, however, $\mathbf{x}$ is a core point and $\mathbf{y}$ is a *border point*, meaning that it is a point on the border of the cluster, direct reachability may *not* be symmetric for them.

   Remark 2. The second criterion in the definition indicates that the point $\mathbf{y}$ is a core point.

6. **Density-reachable Points:** A point $\mathbf{x}$ is *density-reachable* from a point $\mathbf{y}$ with respect to ε and $n_{\min}$ if there exist $m \in \mathbb{N}$ and a chain of points $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_m$, where $\mathbf{y}_1 = \mathbf{y}$ and $\mathbf{y}_m = \mathbf{x}$ such that $\mathbf{y}_{i+1}$ is directly density-reachable from $\mathbf{y}_i$.

   Remark 1. Density reachability is a transitive relation, but is *not* symmetric in general.

   Remark 2. Two border points of the same cluster C may *not* be density-reachable from each other because the core point condition might *not* hold for both of them. However, there must be a core point in C from which both border points are density-reachable.

7. **Density-connected:** A point $\mathbf{x}$ is *density-connected* to a point $\mathbf{y}$ with respect to ε and $n_{\min}$ if there is a point $\mathbf{z}$ such that both $\mathbf{x}$ and $\mathbf{y}$ are density-reachable from $\mathbf{z}$ with respect to ε and $n_{\min}$.

Remark 1. Density-connectivity is a symmetric relation.

Remark 2. For density-reachable points, the relation of density-connectivity is reflexive.

8. **Cluster in DBSCAN:** A *cluster* C with respect to ε and $n_{\min}$ is a non-empty subset of $\mathcal{D}$ satisfying the following two conditions:
   (a) *Maximality:* for any $\mathbf{x}, \mathbf{y} \in \mathcal{D}$, if $\mathbf{x} \in C$ and $\mathbf{y}$ is density-reachable from $\mathbf{x}$ with respect to ε and $n_{\min}$, then $\mathbf{y} \in C$;
   (b) *Connectivity:* For any $\mathbf{x}, \mathbf{y} \in C$, $\mathbf{x}$ is density-connected to $\mathbf{y}$ with respect to ε and $n_{\min}$.

   Remark. A cluster with respect to ε and $n_{\min}$ contains at least $n_{\min}$ points.

9. **Noise in DBSCAN:** Let $C_1, C_2, \cdots, C_K$ be the clusters of $\mathcal{D}$ with respect to parameters ε_k and $n_{\min,k}$, for all $k = 1, 2, \cdots, K$. Then, we define the *noise* as the set of points in $\mathcal{D}$ *not* belonging to any cluster C_k.

10. **Lemmas:**
    (a) Let $\mathbf{x} \in \mathcal{D}$ and $|\mathcal{N}_\varepsilon(\mathbf{x})| > n_{\min}$. Then, the set
    $$S := \left\{ \mathbf{z} \in \mathcal{D} \,\middle|\, \mathbf{z} \text{ is density-reachable from } \mathbf{x} \text{ with respect to } \varepsilon \text{ and } n_{\min} \right\}$$
    is a cluster with respect to ε and $n_{\min}$.
    (b) Let C be a cluster with respect to ε and $n_{\min}$ and let $\mathbf{x}$ be any point in C with $|\mathcal{N}_\varepsilon(\mathbf{x})| > n_{\min}$. Then, $C = S$.

11. **DBSCAN Algorithm:** The complete DBSCAN algorithm is shown in Algorithm 8.

Algorithm 8 DBSCAN Algorithm

Require: Data to be clustered, $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$;
Require: $\varepsilon > 0$ and $n_{\min} > 0$.
1: Choose an arbitrary point $\mathbf{x}$ from $\mathcal{D}$;
2: **if** $\mathbf{x}$ is a core point **then**
3: Retrieve all points that are density-reachable from $\mathbf{x}$ and obtain the cluster containing $\mathbf{x}$, and label the remaining ones as noise;
4: **else**
5: Mark $\mathbf{x}$ as a noise point and go back to Step 1.
6: **end if**
7: Repeat the preceding steps until all remaining points labeled as noises have distances to all clusters by more than ε, where the distance of a point $\mathbf{x}$ to a cluster C is defined to be
$$\min_{\{i \,|\, i \in C\}} d(\mathbf{x}, \mathbf{x}_i).$$

12. Choice of $n_{\min}$:

(a) As a rule of thumb, $n_{\min} = 2 \times$ dimensionality of data can be used;

(b) Domain knowledge and a deep understanding of data can help in choosing the most appropriate value of $n_{\min}$.

13. Choice of ε:

(a) *Effects of ε:*

i. if ε is chosen too small, a large part of the data will not be clustered; but

ii. if ε is chosen too large, clusters will merge and the majority of objects will be in the same cluster.

In general, small values of ε are preferable, and, as a rule of thumb, only a small fraction of points should be within this distance of one another.

(b) *How to Choose ε in Practice:* We create a so-called *K-distance plot*, where $K = n_{\min} - 1$. K-distance plot plots the distances to the K-th nearest neighbor of all points, ordered from the largest to the smallest. Good candidates of ε are those where this plot shows an "elbow".

Remark. The K nearest neighbors of a point consist of exactly $n_{\min}$ points.

14. Advantages:

(a) DBSCAN does *not* require the specification of the number of clusters *a priori*, as opposed to the K-means algorithm or the K-medoids algorithm;

(b) DBSCAN can find *arbitrarily-shaped* clusters. It can even find a cluster completely surrounded by (but not connected to) a different cluster;

(c) DBSCAN has a notion of noise, and is robust to outliers;

(d) DBSCAN requires just two parameters and is mostly insensitive to the ordering of the points in the database.

(e) The parameters ε and $n_{\min}$ can be set by a domain expert, if the data is well understood.

15. Disadvantages:

(a) DBSCAN is *not* entirely deterministic: border points that are reachable from more than one cluster can be part of either cluster, depending on the order the data are processed;

(b) The quality of DBSCAN depends on the dissimilarity measure. This is especially true for high-dimensional data, where the algorithm may suffer from the so-called "curse of dimensionality";

(c) DBSCAN cannot cluster datasets well with large differences in densities, since the ε-$n_{\min}$ combination cannot then be chosen appropriately for *all* clusters;

(d) If the data and scale are not well understood, choosing a meaningful distance threshold ε can be difficult.

VI. Spectral Clustering

1. **Overview:** Spectral clustering is a generalization of standard clustering methods and is able to handle arbitrarily-shaped clusters. It is very easy to implement and can be solved efficiently by standard linear algebra methods.

2. **Setup:** Let

   (a) $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$ be a set of data points to be clustered, and

   (b) $s_{i,i'} := s(\mathbf{x}_i, \mathbf{x}_{i'}) \geq 0$ be the similarity between $\mathbf{x}_i$ and $\mathbf{x}_{i'}$.

   Remark. Similarity measurements $s(\mathbf{x}_i, \mathbf{x}_{i'})$ can be obtained by a transformation of the dissimilarity measurements $d(\mathbf{x}_i, \mathbf{x}_{i'})$.

3. **Similarity Graph:** A *similarity graph*, denoted by $\mathcal{G} = (V, E)$, is an undirected graph with vertex set $V = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$. In addition, $\mathcal{G}$ satisfies the following properties:

   (a) two vertices, say $\mathbf{x}_i$ and $\mathbf{x}_{i'}$, are connected if their similarity $s_{i,i'}$ is strictly positive or is larger than a certain threshold;

   (b) each edge is weighted with the weight being the similarity of the corresponding two vertices.

4. **Reformulation of Clustering Problem:** With the similarity graph defined as above, the clustering can be recast as the following: Given a similarity graph, we want to find a partition of the graph such that

   (a) the edges between different clusters have very low weights, meaning that points in different clusters are dissimilar from each other, and

   (b) the edges within a cluster have high weights, meaning that points within the same cluster are similar to each other.

5. **Basic Definitions from Graph Theory:** Consider a general graph (not necessarily a similarity graph) $\mathcal{G} = (V, E)$, with the vertex set being $V = \{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n\}$ and the edge connecting $\mathbf{v}_i$ and $\mathbf{v}_{i'}$ being weighted by $w_{i,i'}$, for all $i, i' = 1, 2, \cdots, n$.

   (a) *Weighted Adjacency Matrix:* The *weighted adjacency matrix* of the graph is the matrix $\mathbf{W}$ with the (i, i')-th entry being $w_{i,i'}$, for all $i, i' = 1, 2, \cdots, n$.
   Remark 1. If $w_{i,i'} = 0$, this means that the vertices $\mathbf{v}_i$ and $\mathbf{v}_{i'}$ are *not* connected by an edge.
   Remark 2. Since $\mathcal{G}$ is undirected, we require $w_{i,i'} = w_{i',i}$. In other words, $\mathbf{W}$ is a symmetric matrix.

   (b) *Degree:* The *degree* of a vertex $\mathbf{v}_i \in V$ is defined as

   $$d_i := \sum_{i'=1}^{n} w_{i,i'},$$

   i.e., the sum of all weights of edges connected with $\mathbf{v}_i$.

(c) *Degree Matrix:* The *degree matrix* $\mathbf{D} \in \mathbb{R}^{n\times n}$ is defined as the diagonal matrix with the degrees $d_1, d_2, \cdots, d_n$ on the diagonal.

(d) *Connected Subset of Vertices:* Let $A \subset V$ be a subset of vertices in V. Then, A is said to be *connected* if any two vertices in A can be joined by a path such that all intermediate vertices also reside in A.

(e) *Connected Component:* A subset $A \subset V$ is said to be a *connected component* if

i. it is connected, and

ii. there are no connections between vertices in A and $A^{\complement}$.

(f) *Partition:* The nonempty sets $A_1, A_2, \cdots, A_k$ form a *partition* of the graph if $A_i \cap A_{i'} = \emptyset$ for $i \neq i'$ and $A_1 \cup A_2 \cup \cdots \cup A_k = V$.

(g) *Size of a Subset of Vertices:* Let $A \subset V$ be a subset of vertices in V. We use the following two quantities to measure the *size* of A

$$\begin{aligned} |A| &:= \text{ the number of vertices in } A, \\ \mathrm{vol}(A) &:= \sum_{\{i \,|\, \mathbf{v}_i \in A\}} d_i. \end{aligned}$$

Intuitively, $|A|$ measures the size of A by its number of vertices, while $\mathrm{vol}(A)$ measures the size of A by summing over the weights of all edges attached to vertices in A.

6. Notation:

(a) For two not necessarily disjoint sets $A, B \subset V$, we define

$$W(A, B) := \sum_{\{(i,i') | \mathbf{v}_i \in A, \mathbf{v}_{i'} \in B\}} w_{i,i'}.$$

(b) Let A be a subset of vertices in V. Denote its complement by $A^{\complement}$.

(c) Let A be a subset of vertices in V. Let $\mathbf{1}_A := (a_1, a_2, \cdots, a_n)^\top \in \mathbb{R}^n$, where, for all $i = 1, 2, \cdots, n$,

$$a_i = \begin{cases} 1, & \text{if } \mathbf{v}_i \in A, \\ 0, & \text{if } \mathbf{v}_i \notin A. \end{cases}$$

7. Different Similarity Graphs:

(a) *ε-neighborhood Graph:* We connect *all* data points whose pairwise distances are smaller than a pre-specified parameter $\varepsilon > 0$.

Remark. Since the distances between all connected points are roughly of the same scale (at most ε), the ε-neighborhood graph is usually considered as an *unweighted* graph.

(b) *k-nearest Neighbor Graph:* We connect the data point $\mathbf{x}_i$ with the data point $\mathbf{x}_{i'}$ if $\mathbf{x}_{i'}$ is among the k-nearest neighbors of $\mathbf{x}_i$.

Remark. This approach leads to a *directed* graph, as the neighborhood relationship is *not* symmetric. To make this graph undirected, there are two ideas:

i. Ignore the directions of the edges — we connect $\mathbf{x}_i$ and $\mathbf{x}_{i'}$ with an undirected edge if $\mathbf{x}_i$ is among the k-nearest neighbors of $\mathbf{x}_{i'}$ or if $\mathbf{x}_{i'}$ is among the k-nearest neighbors of $\mathbf{x}_i$. Call this resulting graph the k-nearest neighbor graph.

ii. Connect vertices $\mathbf{x}_i$ and $\mathbf{x}_{i'}$ if both $\mathbf{x}_i$ is among the k-nearest neighbors of $\mathbf{x}_{i'}$ and $\mathbf{x}_{i'}$ is also among the k-nearest neighbors of $\mathbf{x}_i$. Call this resulting graph the *mutual k-nearest neighbor graph.*

(c) *Fully Connected Graph:* We connect all points with positive similarity with each other and weigh all edges by $s(\mathbf{x}_i, \mathbf{x}_{i'})$.

This requires us to choose an appropriate similarity function that models local neighborhood. One such choice is the Gaussian similarity function

$$s(\mathbf{x}_i, \mathbf{x}_{i'}) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_{i'}\|_2^2}{2\sigma^2}\right), \tag{10}$$

where $\sigma > 0$ controls the width of the neighborhood.

8. Un-normalized Graph Laplacian Matrix: The *un-normalized graph Laplacian matrix* is defined to be

$$\mathbf{L} := \mathbf{D} - \mathbf{W}, \tag{11}$$

where $\mathbf{D}$ is the degree matrix and $\mathbf{W}$ is the weighted adjacency matrix.

9. Properties of L:

(a) For any vector $\mathbf{a} = (a_1, a_2, \cdots, a_n) \in \mathbb{R}^n$, we have

$$\mathbf{a}^\top \mathbf{L}\mathbf{a} = \frac{1}{2}\sum_{i=1}^{n}\sum_{i'=1}^{n} w_{i,i'}(a_i - a_{i'})^2;$$

(b) $\mathbf{L}$ is symmetric and positive semi-definite;

(c) The smallest eigenvalue of $\mathbf{L}$ is 0 with the corresponding eigenvector being $\mathbf{1}_n$;

(d) $\mathbf{L}$ has n non-negative real-valued eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_n = 0$.

10. Number of Connected Components and Zero Eigenvalue of L: Let $\mathcal{G}$ be an undirected graph with non-negative weights. Then, the following statements hold:

(a) The multiplicity k of the eigenvalue 0 of $\mathbf{L}$ equals the number of connected components $A_1, A_2, \cdots, A_k$ in the graph.

(b) The eigen-space of the eigenvalue 0 is spanned by the indicator vectors $\mathbf{1}_{A_1}, \cdots, \mathbf{1}_{A_k}$ of those components.

11. **Normalized Graph Laplacian Matrix:** There are two kinds of normalized graph Laplacian matrices, namely,

$$\mathbf{L}_{\text{sym}} := \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}} = \mathbf{I}_n - \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}, \tag{12}$$

$$\mathbf{L}_{\text{rw}} := \mathbf{D}^{-1}\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W}. \tag{13}$$

12. **Properties of $\mathbf{L}_{\text{sym}}$ and $\mathbf{L}_{\text{rw}}$:**

    (a) For every $\mathbf{a} := (a_1, a_2, \cdots, a_n)^\top \in \mathbb{R}^n$, we have

$$\mathbf{a}^\top \mathbf{L}_{\text{sym}} \mathbf{a} = \frac{1}{2}\sum_{i=1}^{n}\sum_{i'=1}^{n} w_{i,i'}\left(\frac{a_i}{\sqrt{d_i}} - \frac{a_{i'}}{\sqrt{d_{i'}}}\right)^2.$$

    (b) λ is an eigenvalue of $\mathbf{L}_{\text{rw}}$ with eigenvector $\mathbf{u}$ if and only if λ is an eigenvalue of $\mathbf{L}_{\text{sym}}$ with eigenvector $\mathbf{w} = \mathbf{D}^{-\frac{1}{2}}\mathbf{u}$;

    (c) λ is an eigenvalue of $\mathbf{L}_{\text{rw}}$ with eigenvector $\mathbf{u}$ if and only if λ and $\mathbf{u}$ solve the generalized eigen-problem $\mathbf{L}\mathbf{u} = \lambda\mathbf{D}\mathbf{u}$;

    (d) Zero is an eigenvalue of $\mathbf{L}_{\text{rw}}$ with the eigenvector $\mathbf{1}_n$, and zero is an eigenvalue of $\mathbf{L}_{\text{sym}}$ with eigenvector $\mathbf{D}^{-\frac{1}{2}}\mathbf{1}_n$;

    (e) $\mathbf{L}_{\text{sym}}$ and $\mathbf{L}_{\text{rw}}$ are positive semi-definite and have n non-negative real-valued eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n = 0$.

13. **Number of Connected Components and Spectra of $\mathbf{L}_{\text{sym}}$ and $\mathbf{L}_{\text{rw}}$**: Let $\mathcal{G}$ be an undirected graph with non-negative weights. Then, the multiplicity k of the eigenvalue 0 of both $\mathbf{L}_{\text{rw}}$ and $\mathbf{L}_{\text{sym}}$ equals the number of connected components $A_1, A_2, \cdots, A_k$ in the graph. Furthermore,

    (a) for $\mathbf{L}_{\text{rw}}$, the eigen-space of 0 is spanned by the indicator vectors $\mathbf{1}_{A_i}$'s;

    (b) for $\mathbf{L}_{\text{sym}}$, the eigen-space of 0 is spanned by the vectors $\mathbf{D}^{-\frac{1}{2}}\mathbf{1}_{A_i}$'s.

14. **Spectral Clustering Algorithm — Using Un-normalized Graph Laplacian L:** The complete spectral clustering algorithm using un-normalized graph Laplacian $\mathbf{L}$ is shown in Algorithm 9.

Algorithm 9 Spectral Clustering Algorithm (using $\mathbf{L}$)

Require: Similarity matrix, $\mathbf{S} \in \mathbb{R}^{n\times n}$;
Require: The number of clusters, K.
1: Construct a similarity graph and let $\mathbf{W}$ be its weighted adjacency matrix;
2: Compute the un-normalized graph Laplacian matrix $\mathbf{L}$;
3: Compute the eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_K$ associated with the K smallest eigenvalues of $\mathbf{L}$;
4: Let $\mathbf{U} \in \mathbb{R}^{n\times K}$ be the matrix containing the vectors $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_K$ as columns;
5: For $i = 1, 2, \cdots, n$, let $\mathbf{y}_i \in \mathbb{R}^K$ be the vector corresponding to the i-th row of $\mathbf{U}$;
6: Cluster the points $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n \in \mathbb{R}^K$ with the K-means clustering algorithm into clusters $C_1, C_2, \cdots, C_K$.
7: **return** Clusters $A_1, A_2, \cdots, A_K$ with $A_k = \{i \,|\, \mathbf{y}_i \in C_k\}$ for all $k = 1, 2, \cdots, K$.

15. Spectral Clustering Algorithm — Using Normalized Graph Laplacian $\mathbf{L}_{\text{rw}}$: The complete spectral clustering algorithm using normalized graph Laplacian $\mathbf{L}_{\text{rw}}$ is shown in Algorithm 10.

Algorithm 10 Spectral Clustering Algorithm (using $\mathbf{L}_{\text{rw}}$)

Require: Similarity matrix, $\mathbf{S} \in \mathbb{R}^{n\times n}$;
Require: The number of clusters, K.
1: Construct a similarity graph and let $\mathbf{W}$ be its weighted adjacency matrix;
2: Compute the un-normalized graph Laplacian matrix $\mathbf{L}$;
3: Compute the eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_K$ associated with the K smallest eigenvalues of the generalized eigen-problem $\mathbf{Lu} = \lambda\mathbf{Du}$;
4: Let $\mathbf{U} \in \mathbb{R}^{n\times K}$ be the matrix containing the vectors $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_K$ as columns;
5: For $i = 1, 2, \cdots, n$, let $\mathbf{y}_i \in \mathbb{R}^K$ be the vector corresponding to the i-th row of $\mathbf{U}$;
6: Cluster the points $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n \in \mathbb{R}^K$ with the K-means clustering algorithm into clusters $C_1, C_2, \cdots, C_K$.
7: **return** Clusters $A_1, A_2, \cdots, A_K$ with $A_k = \{i \,|\, \mathbf{y}_i \in C_k\}$ for all $k = 1, 2, \cdots, K$.

Remark. Algorithm 10 uses the generalized eigenvectors of $\mathbf{L}$, which corresponds to the eigenvectors of the normalized graph Laplacian matrix $\mathbf{L}_{\text{rw}}$.

16. Spectral Clustering Algorithm — Using normalized graph Laplacian $\mathbf{L}_{\text{sym}}$: The complete spectral clustering algorithm using normalized graph Laplacian $\mathbf{L}_{\text{sym}}$ is shown in Algorithm 11.

Algorithm 11 Spectral Clustering Algorithm (using $\mathbf{L}_{\text{sym}}$)

Require: Similarity matrix, $\mathbf{S} \in \mathbb{R}^{n\times n}$;
Require: The number of clusters, K.
1: Construct a similarity graph and let $\mathbf{W}$ be its weighted adjacency matrix;
2: Compute the un-normalized graph Laplacian matrix $\mathbf{L}$;
3: Compute the eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_K$ associated with the K smallest eigenvalues of the normalized graph Laplacian matrix $\mathbf{L}_{\text{sym}}$;
4: Let $\mathbf{U} \in \mathbb{R}^{n\times K}$ be the matrix containing the vectors $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_K$ as columns;
5: Form the matrix $\mathbf{T} \in \mathbb{R}^{n\times K}$ from $\mathbf{U}$ by normalizing the rows to norm 1, i.e.,

$$t_{i,k} = \frac{u_{i,k}}{\sqrt{\sum_{\ell=1}^{K} u_{i,\ell}^2}}, \qquad \text{for all } i = 1, 2, \cdots, n \text{ and } k = 1, 2, \cdots, K;$$

6: For $i = 1, 2, \cdots, n$, let $\mathbf{y}_i \in \mathbb{R}^K$ be the vector corresponding to the i-th row of $\mathbf{T}$;
7: Cluster the points $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n \in \mathbb{R}^K$ with the K-means clustering algorithm into clusters $C_1, C_2, \cdots, C_K$.
8: **return** Clusters $A_1, A_2, \cdots, A_K$ with $A_k = \{i \,|\, \mathbf{y}_i \in C_k\}$ for all $k = 1, 2, \cdots, K$.

17. Graph Cut Point of View of Spectral Clustering:

(a) *Motivation:* Given a similarity graph $\mathcal{G} = (V, E)$ constructed from the data $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n$, clustering problem can be reformulated as the following: we want to find a partition of the graph such that

i. the edges between different groups have very low weights, meaning that points in different clusters are dissimilar from each other, and

ii. the edges within a group have high weights, meaning that points within the same cluster are similar to each other.

(b) *Mincut Formulation:* Given a similarity graph $\mathcal{G}$ and the corresponding weighted adjacency matrix $\mathbf{W} \in \mathbb{R}^{n\times n}$, the motivation above can be naturally translated a *mincut problem*.

More precisely, for a given number of K subsets, we choose a partition $A_1, A_2, \cdots, A_K$ to minimize the following cut objective function

$$\text{Cut}\,(A_1, A_2, \cdots, A_K) := \frac{1}{2}\sum_{k=1}^{K} W(A_K, A_k^{\complement}). \tag{14}$$

Remark. The issue with this formulation is that, typically, the solution separates an individual vertex from the rest of the graph, which is obviously unsatisfactory.

(c) *Improvement Upon Mincut Problem:* We require that the subsets $A_1, A_2, \cdots, A_K$ are all "reasonably large". Two objective functions that meet this requirement are

$$\text{RatioCut}\,(A_1, A_2, \cdots, A_K) := \frac{1}{2}\sum_{k=1}^{K} \frac{W(A_k, A_k^{\complement})}{|A_k|} = \sum_{k=1}^{K} \frac{\text{Cut}(A_k, A_k^{\complement})}{|A_k|}, \tag{15}$$

$$\text{NCut}\,(A_1, A_2, \cdots, A_K) := \frac{1}{2}\sum_{k=1}^{K} \frac{W(A_k, A_k^{\complement})}{\text{vol}(A_k)} = \sum_{k=1}^{K} \frac{\text{Cut}(A_k, A_k^{\complement})}{\text{vol}(A_k)}. \tag{16}$$

Remark 1. Both objective functions above take a relatively small value if the clusters A_k's are *not* too small.

Remark 2. The minimum of the function $\sum_{k=1}^{K} 1/|A_k|$ is achieved if

$$|A_1| = |A_2| = \cdots = |A_K|,$$

and the minimum of the function $\sum_{k=1}^{K} 1/\text{vol}(A_k)$ is achieved if

$$\text{vol}(A_1) = \text{vol}(A_2) = \cdots = \text{vol}(A_K).$$

Hence, what the objective functions RatioCut and NCut attempt to achieve is that the clusters are "balanced", as measured by the number of vertices or edge weights, respectively.

Remark 3. Minimizing RatioCut and NCut are both NP-hard. Spectral clustering is a way to solve relaxed versions of these problems.

(d) *Minimizing* RatioCut *When* $K = 2$*:* Let $K = 2$. We consider minimizing the RatioCut objective function, i.e.,

$$\underset{A \subset V}{\text{minimize}}\ \text{RatioCut}(A, A^{\complement}). \tag{17}$$

Let $\mathbf{A} \subset V$ be given and $\mathbf{a} := (a_1, a_2, \cdots, a_n)^\top \in \mathbb{R}^n$ with

$$a_i = \begin{cases} \sqrt{|A^{\complement}|/|A|}, & \text{if } \mathbf{x}_i \in A, \\ -\sqrt{|A|/|A^{\complement}|}, & \text{if } \mathbf{x}_i \notin A, \end{cases} \tag{18}$$

for all $i = 1, 2, \cdots, n$. Then,

$$\begin{aligned}
\mathbf{a}^\top \mathbf{L}\mathbf{a} &= \frac{1}{2}\sum_{i=1}^{n}\sum_{i'=1}^{n} w_{i,i'}(a_i - a_{i'})^2 \\
&= \frac{1}{2}\sum_{\{(i,i')\,|\,\mathbf{x}_i \in A, \mathbf{x}_{i'} \in A^{\complement}\}} w_{i,i'}\left(\sqrt{\frac{|A^{\complement}|}{|A|}} + \sqrt{\frac{|A|}{|A^{\complement}|}}\right)^2 \\
&\qquad + \frac{1}{2}\sum_{\{(i,i')\,|\,\mathbf{x}_i \in A^{\complement}, \mathbf{x}_{i'} \in A\}} w_{i,i'}\left(-\sqrt{\frac{|A|}{|A^{\complement}|}} - \sqrt{\frac{|A^{\complement}|}{|A|}}\right)^2 \\
&= \frac{1}{2}W(A, A^{\complement})\left(\sqrt{\frac{|A^{\complement}|}{|A|}} + \sqrt{\frac{|A|}{|A^{\complement}|}}\right)^2 + \frac{1}{2}W(A^{\complement}, A)\left(-\sqrt{\frac{|A|}{|A^{\complement}|}} - \sqrt{\frac{|A^{\complement}|}{|A|}}\right)^2 \\
&= W(A, A^{\complement})\left(\sqrt{\frac{|A^{\complement}|}{|A|}} + \sqrt{\frac{|A|}{|A^{\complement}|}}\right)^2 \\
&= \text{Cut}(A, A^{\complement})\left(\frac{|A^{\complement}|}{|A|} + \frac{|A|}{|A^{\complement}|} + 2\right) \\
&= \text{Cut}(A, A^{\complement})\left(\frac{|A^{\complement}| + |A|}{|A|} + \frac{|A^{\complement}| + |A|}{|A^{\complement}|}\right) \\
&= |V| \cdot \text{RatioCut}\,(A, A^{\complement}).
\end{aligned}$$

Additionally, $\mathbf{a}$ defined above is orthogonal to the vector $\mathbf{1}_n$, since

$$\begin{aligned}
\mathbf{1}_n^\top \mathbf{a} = \sum_{i=1}^{n} a_i &= \sum_{\{i\,|\,\mathbf{x}_i \in A\}} \sqrt{\frac{|A^{\complement}|}{|A|}} - \sum_{\{i\,|\,\mathbf{x}_i \in A^{\complement}\}} \sqrt{\frac{|A|}{|A^{\complement}|}} \\
&= |A|\sqrt{\frac{|A^{\complement}|}{|A|}} - |A^{\complement}|\sqrt{\frac{|A|}{|A^{\complement}|}} \\
&= \sqrt{|A||A^{\complement}|} - \sqrt{|A||A^{\complement}|} \\
&= 0.
\end{aligned}$$

Finally, we have

$$\begin{aligned}\|\mathbf{a}\|_2^2 = \sum_{i=1}^n a_i^2 &= \sum_{\{i\,|\,\mathbf{x}_i \in A\}} \frac{|A^{\complement}|}{|A|} + \sum_{\{i\,|\,\mathbf{x}_i \in A^{\complement}\}} \frac{|A|}{|A^{\complement}|} \\ &= |A|\frac{|A^{\complement}|}{|A|} + |A^{\complement}|\frac{|A|}{|A^{\complement}|} \\ &= |A^{\complement}| + |A| \\ &= n.\end{aligned}$$

Therefore, the problem (17) is equivalent to the following one

$$\begin{aligned}\underset{A \subset V}{\text{minimize}}\ & \mathbf{a}^\top \mathbf{L}\mathbf{a} \\ \text{subject to}\ & \mathbf{1}_n^\top \mathbf{a} = 0, \\ & \text{components of } \mathbf{a} \text{ are of the form in (18), and} \\ & \|\mathbf{a}\|_2 = \sqrt{n}.\end{aligned}$$

This preceding problem is still NP-hard, since components of $\mathbf{a}$ are only allowed two values. Instead, we consider the following relaxed problem

$$\begin{aligned}\underset{\mathbf{a} \in \mathbb{R}^n}{\text{minimize}}\ & \mathbf{a}^\top \mathbf{L}\mathbf{a} \\ \text{subject to}\ & \mathbf{1}_n^\top \mathbf{a} = 0, \\ & \|\mathbf{a}\|_2 = \sqrt{n}.\end{aligned}$$

The solution to this problem is given by the eigenvector associated with the second smallest eigenvalue of $\mathbf{L}$. Letting $\hat{\mathbf{a}} = (\hat{a}_1, \hat{a}_2, \cdots, \hat{a}_n)^\top \in \mathbb{R}^n$ be the solution, we can assign each data point to a cluster by

$$\begin{cases}\mathbf{x}_i \in A, & \text{if } \hat{a}_i \geq 0, \\ \mathbf{x}_i \in A^{\complement}, & \text{if } \hat{a}_i < 0.\end{cases} \tag{19}$$

Connection to Spectral Clustering: In spectral clustering, we consider the coordinates a_i as points in $\mathbb{R}$ and cluster them into two groups by the K-means clustering algorithm. Then, we carry over the resulting clustering to the underlying data points

$$\begin{cases}\mathbf{x}_i \in A, & \text{if } a_i \in C, \\ \mathbf{x}_i \in A^{\complement}, & \text{if } a_i \in C^{\complement}.\end{cases}$$

This is the spectral clustering algorithm using the un-normalized graph Laplacian matrix when $K = 2$.

(e) *Minimizing* RatioCut *When* $K > 2$*:* Let $K > 2$ be given. Given a partition of V into K subsets, denoted by $A_1, A_2, \cdots, A_K$, we define K indicator vectors $\mathbf{h}_k := (h_{k,1}, h_{k,2}, \cdots, h_{k,n})^\top \in \mathbb{R}^n$ by

$$h_{k,i} = \begin{cases} |A_k|^{-\frac{1}{2}}, & \text{if } \mathbf{x}_i \in A_k, \\ 0, & \text{otherwise}, \end{cases} \tag{20}$$

for all $i = 1, 2, \cdots, n$ and $k = 1, 2, \cdots, K$. Let $\mathbf{H} \in \mathbb{R}^{n \times K}$ be the matrix containing these K indicator vectors as columns. Notice that columns of $\mathbf{H}$ are orthonormal, i.e., $\mathbf{H}^\top \mathbf{H} = \mathbf{I}_K$.

By a calculation as before, we have, for all $k = 1, 2, \cdots, K$,

$$\mathbf{h}_k^\top \mathbf{L} \mathbf{h}_k = \frac{\mathrm{Cut}(A_k, A_k^\complement)}{|A_k|}$$

and

$$\mathbf{h}_k^\top \mathbf{L} \mathbf{h}_k = [\mathbf{H}^\top \mathbf{L} \mathbf{H}]_{k,k},$$

where $[\mathbf{A}]_{k,k}$ indicates the (k,k)-th entry of the matrix $\mathbf{A}$.

Combining everything above, we have

$$\text{RatioCut}\,(A_1, A_2, \cdots, A_K) = \sum_{k=1}^{K} \mathbf{h}_k^\top \mathbf{L} \mathbf{h}_k = \sum_{k=1}^{K} [\mathbf{H}^\top \mathbf{L} \mathbf{H}]_{k,k} = \mathrm{trace}(\mathbf{H}^\top \mathbf{L} \mathbf{H}).$$

Therefore, the problem of minimizing $\text{RatioCut}\,(A_1, A_2, \cdots, A_K)$ can be rewritten as

$$\begin{aligned} &\underset{A_1, A_2, \cdots, A_K}{\text{minimize}} \ \mathrm{trace}(\mathbf{H}^\top \mathbf{L} \mathbf{H}) \\ &\text{subject to } \ \mathbf{H}^\top \mathbf{H} = \mathbf{I}_K, \\ &\qquad\qquad \text{entries of } \mathbf{H} \text{ are given by (20).} \end{aligned}$$

Similar to to before, we relax the preceding problem to the following one

$$\begin{aligned} &\underset{\mathbf{H} \in \mathbb{R}^{n \times K}}{\text{minimize}} \ \mathrm{trace}(\mathbf{H}^\top \mathbf{L} \mathbf{H}) \\ &\text{subject to } \ \mathbf{H}^\top \mathbf{H} = \mathbf{I}_K. \end{aligned}$$

The solution then is given the matrix of dimensionality $n \times K$ containing K eigenvectors of $\mathbf{L}$ associated with the smallest K eigenvalues.

Remark. There is *no* guarantee on the quality of the solution of the relaxed problem compared to the exact solution.

More precisely, if $A_1, A_2, \cdots, A_K$ is the exact solution of minimizing RatioCut, and $B_1, B_2, \cdots, B_K$ is the solution constructed by unnormalized spectral clustering, then

$$\text{RatioCut}\,(B_1, B_2, \cdots, B_K) - \text{RatioCut}\,(A_1, A_2, \cdots, A_K)$$

can be arbitrary large.

(f) *Minimizing* NCut *When $K = 2$:* Let $K = 2$, and A and $A^{\complement}$ be a partition of V. Define the indicator vector $\mathbf{a} = (a_1, a_2, \cdots, a_n) \in \mathbb{R}^n$ to be

$$a_i = \begin{cases} \sqrt{\mathrm{vol}(A^{\complement})/\mathrm{vol}(A)}, & \text{if } \mathbf{x}_i \in A, \\ -\sqrt{\mathrm{vol}(A)/\mathrm{vol}(A^{\complement})}, & \text{if } \mathbf{x}_i \in A^{\complement}. \end{cases} \tag{21}$$

Then, we have

$$\mathbf{Da} = \begin{pmatrix} d_1 a_1 \\ d_2 a_2 \\ \vdots \\ d_n a_n \end{pmatrix} \in \mathbb{R}^n.$$

As a consequence, we have

$$\begin{aligned} (\mathbf{Da})^{\top} \mathbf{1}_n &= \sum_{i=1}^{n} d_i a_i \\ &= \sum_{\{i \,|\, \mathbf{x}_i \in A\}} d_i a_i + \sum_{\{i \,|\, \mathbf{x}_i \in A^{\complement}\}} d_i a_i \\ &= \sum_{\{i \,|\, \mathbf{x}_i \in A\}} d_i \sqrt{\frac{\mathrm{vol}(A^{\complement})}{\mathrm{vol}(A)}} - \sum_{\{i \,|\, \mathbf{x}_i \in A^{\complement}\}} d_i \sqrt{\frac{\mathrm{vol}(A)}{\mathrm{vol}(A^{\complement})}} \\ &= \mathrm{vol}(A) \sqrt{\frac{\mathrm{vol}(A^{\complement})}{\mathrm{vol}(A)}} - \mathrm{vol}(A^{\complement}) \sqrt{\frac{\mathrm{vol}(A)}{\mathrm{vol}(A^{\complement})}} \\ &= \sqrt{\mathrm{vol}(A) \times \mathrm{vol}(A^{\complement})} - \sqrt{\mathrm{vol}(A) \times \mathrm{vol}(A^{\complement})} \\ &= 0, \end{aligned}$$

and

$$\begin{aligned} \mathbf{a}^{\top} \mathbf{Da} &= \sum_{i=1}^{n} d_i a_i^2 \\ &= \sum_{\{i \,|\, \mathbf{x}_i \in A\}} d_i a_i^2 + \sum_{\{i \,|\, \mathbf{x}_i \in A^{\complement}\}} d_i a_i^2 \\ &= \sum_{\{i \,|\, \mathbf{x}_i \in A\}} d_i \frac{\mathrm{vol}(A^{\complement})}{\mathrm{vol}(A)} + \sum_{\{i \,|\, \mathbf{x}_i \in A^{\complement}\}} d_i \frac{\mathrm{vol}(A)}{\mathrm{vol}(A^{\complement})} \\ &= \mathrm{vol}(A) \frac{\mathrm{vol}(A^{\complement})}{\mathrm{vol}(A)} + \mathrm{vol}(A^{\complement}) \frac{\mathrm{vol}(A)}{\mathrm{vol}(A^{\complement})} \\ &= \mathrm{vol}(V). \end{aligned}$$

Finally, we also have

$$\begin{aligned}
\mathbf{a}^\top \mathbf{L}\mathbf{a} &= \frac{1}{2}\sum_{i=1}^{n}\sum_{i'=1}^{n} w_{i,i'}(a_i - a_{i'})^2 \\
&= \frac{1}{2}\sum_{\{(i,i')\,|\,\mathbf{x}_i\in A, \mathbf{x}_{i'}\in A^{\complement}\}} w_{i,i'}(a_i - a_{i'})^2 + \frac{1}{2}\sum_{\{(i,i')\,|\,\mathbf{x}_i\in A^{\complement}, \mathbf{x}_{i'}\in A\}} w_{i,i'}(a_i - a_{i'})^2 \\
&= \frac{1}{2}\sum_{\{(i,i')\,|\,\mathbf{x}_i\in A, \mathbf{x}_{i'}\in A^{\complement}\}} w_{i,i'}\left(\sqrt{\frac{\mathrm{vol}(A^{\complement})}{\mathrm{vol}(A)}} + \sqrt{\frac{\mathrm{vol}(A)}{\mathrm{vol}(A^{\complement})}}\right)^2 \\
&\quad + \frac{1}{2}\sum_{\{(i,i')\,|\,\mathbf{x}_i\in A^{\complement}, \mathbf{x}_{i'}\in A\}} w_{i,i'}\left(-\sqrt{\frac{\mathrm{vol}(A)}{\mathrm{vol}(A^{\complement})}} - \sqrt{\frac{\mathrm{vol}(A^{\complement})}{\mathrm{vol}(A)}}\right)^2 \\
&= \frac{1}{2}(W(A, A^{\complement}) + W(A^{\complement}, A))\left(\frac{\mathrm{vol}(A^{\complement})}{\mathrm{vol}(A)} + \frac{\mathrm{vol}(A)}{\mathrm{vol}(A^{\complement})} + 2\right) \\
&= \mathrm{Cut}(A, A^{\complement})\left(\frac{\mathrm{vol}(A^{\complement}) + \mathrm{vol}(A)}{\mathrm{vol}(A)} + \frac{\mathrm{vol}(A) + \mathrm{vol}(A^{\complement})}{\mathrm{vol}(A^{\complement})}\right) \\
&= \mathrm{vol}(V)\left(\frac{\mathrm{Cut}(A, A^{\complement})}{\mathrm{vol}(A)} + \frac{\mathrm{Cut}(A, A^{\complement})}{\mathrm{vol}(A^{\complement})}\right) \\
&= \mathrm{vol}(V)\mathrm{NCut}(A, A^{\complement}).
\end{aligned}$$

Therefore, we can write the problem of minimizing $\mathrm{NCut}(A, A^{\complement})$ as

$$\begin{aligned}
&\underset{A\subset V}{\text{minimize}} && \mathbf{a}^\top \mathbf{L}\mathbf{a} \\
&\text{subject to} && (\mathbf{D}\mathbf{a})^\top \mathbf{1}_n = 0, \\
& && \text{entries of } \mathbf{a} \text{ is of the form in (21)}, \\
& && \mathbf{a}^\top \mathbf{D}\mathbf{a} = \mathrm{vol}(V),
\end{aligned} \tag{22}$$

which is NP-hard. Again, we can relax the preceding problem as the following one

$$\begin{aligned}
&\underset{\mathbf{a}\in\mathbb{R}^n}{\text{minimize}} && \mathbf{a}^\top \mathbf{L}\mathbf{a} \\
&\text{subject to} && (\mathbf{D}\mathbf{a})^\top \mathbf{1}_n = 0, \text{ and } \mathbf{a}^\top \mathbf{D}\mathbf{a} = \mathrm{vol}(V).
\end{aligned} \tag{23}$$

If we let $\mathbf{b} := \mathbf{D}^{\frac{1}{2}}\mathbf{a}$, the preceding problem can be rewritten as

$$\begin{aligned}
&\underset{\mathbf{b}\in\mathbb{R}^n}{\text{minimize}} && \mathbf{b}^\top \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}\mathbf{b} \\
&\text{subject to} && \mathbf{b}^\top \mathbf{D}^{\frac{1}{2}}\mathbf{1}_n = 0, \text{ and } \|\mathbf{b}\|_2^2 = \mathrm{vol}(V).
\end{aligned} \tag{24}$$

Notice that $\mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}} = \mathbf{L}_{\mathrm{sym}}$, $\mathbf{D}^{\frac{1}{2}}\mathbf{1}_n$ is the eigenvector of $\mathbf{L}_{\mathrm{sym}}$ associated with its smallest eigenvalue 0, and $\mathrm{vol}(V)$ is a constant. Then, the minimizer of the problem (24) is given by the eigenvector associated with the second smallest eigenvalue

of $\mathbf{L}_{\text{sym}}$. As a consequence, the minimizer to (23) is $\mathbf{D}^{-\frac{1}{2}}$ times the eigenvector associated with the second smallest eigenvalue of $\mathbf{L}_{\text{sym}}$, which is also the eigenvector associated with the second smallest eigenvalue of $\mathbf{L}_{\text{rw}}$.

(g) *Minimizing* NCut *When $K > 2$:* Let $K > 2$ be given. Define the indicator vector $\mathbf{h}_k := (h_{k,1}, h_{k,2}, \cdots, h_{k,n})^\top \in \mathbb{R}^n$ by

$$h_{k,i} = \begin{cases} [\text{vol}(A_k)]^{-\frac{1}{2}}, & \text{if } \mathbf{x}_i \in A_k, \\ 0, & \text{otherwise,} \end{cases} \tag{25}$$

for all $i = 1, 2, \cdots, n$ and $k = 1, 2, \cdots, K$. Let $\mathbf{H} \in \mathbb{R}^{n \times K}$ be the matrix containing these K indicator vectors as columns. Observe that $\mathbf{H}^\top \mathbf{H} = \mathbf{I}_K$, and, for all $k = 1, 2, \cdots, K$,

$$\mathbf{h}_k \mathbf{D} \mathbf{h}_k = 1, \qquad \text{and} \qquad \mathbf{h}_k \mathbf{L} \mathbf{h}_k = \frac{\text{Cut}\,(A_k, A_k^{\complement})}{\text{vol}(A_k)}.$$

Therefore, we can write the problem of minimizing Ncut as

$$\begin{aligned} \underset{A_1, A_2, \cdots, A_K}{\text{minimize}} \quad & \text{trace}(\mathbf{H}^\top \mathbf{L} \mathbf{H}) \\ \text{subject to} \quad & \mathbf{H}^\top \mathbf{D} \mathbf{H} = \mathbf{I}_K, \\ & \text{entries of } \mathbf{H} \text{ is of the form in (25).} \end{aligned}$$

Relaxing the discreteness condition and substituting $\mathbf{T} := \mathbf{D}^{\frac{1}{2}} \mathbf{H}$, we obtain the relaxed problem

$$\begin{aligned} \underset{\mathbf{T} \in \mathbb{R}^{n \times K}}{\text{minimize}} \quad & \text{trace}(\mathbf{T}^\top \mathbf{D}^{-\frac{1}{2}} \mathbf{L} \mathbf{D}^{-\frac{1}{2}} \mathbf{T}) \\ \text{subject to} \quad & \mathbf{T}^\top \mathbf{T} = \mathbf{I}_K. \end{aligned}$$

This, again, is the standard trace minimization problem which is solved by the matrix $\mathbf{T}$ which contains the eigenvectors of $\mathbf{L}_{\text{sym}}$ associated with the K smallest eigenvalues as columns. Resubstituting $\mathbf{H} = \mathbf{D}^{-\frac{1}{2}} \mathbf{T}$, we see that the solution $\mathbf{H}$ consists of the eigenvectors of the matrix $\mathbf{L}_{\text{rw}}$ associated with the K smallest eigenvalues.

18. Random Walk Point of View of Spectral Clustering:

(a) *Overview:* From the point of view of random walks, spectral clustering can be interpreted as trying to find a partition of the graph such that the random walk stays long within the same cluster and seldom jumps between clusters.

(b) *Random Walk on a Graph:* A *random walk on a graph* is a stochastic process which randomly jumps from a vertex to another.

(c) *Transition Probability:* The *transition probability* of jumping in one step from the vertex $\mathbf{x}_i$ to the vertex $\mathbf{x}_{i'}$ is proportional to the edge weight $w_{i,i'}$ and is given by

$$p_{i,i'} := \frac{w_{i,i'}}{d_i}.$$

The *transition matrix* $\mathbf{P} := (p_{i,i'})_{i,i'=1,2,\cdots,n}$ of the random walk is thus defined by

$$\mathbf{P} = \mathbf{D}^{-1}\mathbf{W}.$$

(d) *Property:* If the graph is connected and non-bipartite, then the random walk always possesses a unique stationary distribution

$$\boldsymbol{\pi} := (\pi_1, \pi_2, \cdots, \pi_n)^\top \in \mathbb{R}^n,$$

where $\pi_i = \frac{d_i}{\mathrm{vol}(V)}$.

(e) *Relationship between* $\mathbf{L}_{\mathrm{rw}}$ *and* $\mathbf{P}$*:* Since $\mathbf{L}_{\mathrm{rw}} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W}$, we have

$$\mathbf{L}_{\mathrm{rw}} = \mathbf{I} - \mathbf{P}.$$

As a consequence, λ is an eigenvalue of $\mathbf{L}_{\mathrm{rw}}$ with the eigenvector $\mathbf{u}$ if and only if $1 - \lambda$ is an eigenvalue of $\mathbf{P}$ with the eigenvector $\mathbf{u}$.

(f) NCut *via Transition Probabilities:* Let $\mathcal{G}$ be connected and non-bipartite. Assume that we run the random walk $\{X_t\}_{t\in\mathbb{N}}$ starting with X_0 in the stationary distribution $\boldsymbol{\pi}$. For disjoint subsets $A, B \subset V$, denote by $\mathbb{P}(B \,|\, A) := \mathbb{P}(X_1 \in B \,|\, X_0 \in A)$. Then,

$$\mathrm{NCut}\,(A, A^{\complement}) = \mathbb{P}(A^{\complement} \,|\, A) + \mathbb{P}(A \,|\, A^{\complement}).$$

Proof. First note that

$$\begin{aligned}
\mathbb{P}(X_0 \in A, X_1 \in B) &= \sum_{\{(i,i') \,|\, \mathbf{x}_i \in A, \mathbf{x}_{i'} \in B\}} \mathbb{P}(X_0 = \mathbf{x}_i, X_1 = \mathbf{x}_{i'}) \\
&= \sum_{\{(i,i') \,|\, \mathbf{x}_i \in A, \mathbf{x}_{i'} \in B\}} \pi_i p_{i,i'} \\
&= \sum_{\{(i,i') \,|\, \mathbf{x}_i \in A, \mathbf{x}_{i'} \in B\}} \frac{d_i}{\mathrm{vol}(V)} \frac{w_{i,i'}}{d_i} \\
&= \frac{1}{\mathrm{vol}(V)} \sum_{\{(i,i') \,|\, \mathbf{x}_i \in A, \mathbf{x}_{i'} \in B\}} w_{i,i'}.
\end{aligned}$$

Then, we have

$$\begin{aligned}
\mathbb{P}(X_1 \in B \,|\, X_0 \in A) &= \frac{\mathbb{P}(X_0 \in A, X_1 \in B)}{\mathbb{P}(X_0 \in A)} \\
&= \left(\frac{1}{\mathrm{vol}(V)} \sum_{\{(i,i') \,|\, \mathbf{x}_i \in A, \mathbf{x}_{i'} \in B\}} w_{i,i'} \right) \left(\frac{\mathrm{vol}(A)}{\mathrm{vol}(V)} \right)^{-1} \\
&= \frac{1}{\mathrm{vol}(A)} \sum_{\{(i,i') \,|\, \mathbf{x}_i \in A, \mathbf{x}_{i'} \in B\}} w_{i,i'}.
\end{aligned}$$

The desired result follows from the definition of NCut. ∎

19. **How to Choose the Number of Clusters K:** We adopt the following *eigen-gap heuristic* to choose the number of clusters in spectral clustering.

    Eigen-gap Heuristic: We choose the number K such that all eigenvalues $\lambda_1, \lambda_2, \cdots, \lambda_K$ are very small, but λ_{K+1} is relatively large.

20. **Practical Issues 1 — Choice of Similarity Function:** Similarity function is used to construct a similarity graph, based on which spectral clustering is performed.

    (a) We need to make sure that the local neighborhoods induced by this similarity function are "meaningful"; that is, we need to be sure that points considered to be "very similar" by the similarity function are also closely related in the application;

    (b) Ultimately, the choice of the similarity function depends on the domain the data comes from and the application.

21. **Practical Issues 2 — Type of Similarity Graph:**

    (a) When variables are of different scales, it may be hard to find an appropriate ε value in constructing the ε-neighborhood graph.

    (b) The k-nearest neighbor graph can connect points on different scales.

    (c) The mutual k-nearest neighbor graph

    - works well with the variables of different scales, and
    - has the property that it tends to connect points within regions of constant density, but does *not* connect regions of different densities with each other.

    Hence, the mutual k-nearest neighbor graph seems particularly well-suited if we want to detect clusters of different densities.

    (d) The fully-connected similarity graph based on the Gaussian similarity function (10) leads to a dense similarity matrix, but all others can lead to a sparse similarity matrix.

22. **Practical Issues 3 — Choice of Graph Laplacian:**

    (a) *Overview:* In spectral clustering, one should always look at the degree distribution of the similarity graph.

    - If the graph is very regular and most vertices have approximately the same degree, then all the Laplacians are very similar to each other, and will work equally well for clustering.
    - If the degrees in the graph are very broadly distributed, then the Laplacians differ considerably.

    (b) *Rule of Thumb:*

    i. Between un-normalized and normalized graph Laplacian matrices, the normalized ones are preferred over the unnormalized one;

    ii. Between $\mathbf{L}_{\text{sym}}$ and $\mathbf{L}_{\text{rw}}$ in the normalized case, $\mathbf{L}_{\text{rw}}$ is preferred over $\mathbf{L}_{\text{sym}}$.

(c) *Review of the Objectives in Clustering:* Consider the special case where $K = 2$. Clustering has the following two objectives:

i. We want to find a partition such that points in different clusters are dissimilar to each other, meaning that we want to minimize the between-cluster similarity. In the graph setting, this means to minimize $\text{Cut}(A, A^{\complement})$.

ii. We want to find a partition such that points in the same cluster are similar to each other; that is, we want to maximize the within-cluster similarities $W(A, A)$ and $W(A^{\complement}, A^{\complement})$.

(d) *Why Prefer the Normalized Graph Laplacian Matrix Over the Un-normalized One?:* The spectral clustering using the normalized graph Laplacian matrix implements both clustering objectives above, but that using the unnormalized one *only* implements the first objective.

(e) *Why Prefer* $\mathbf{L}_{\text{rw}}$ *over* $\mathbf{L}_{\text{sym}}$ *?:* The eigenvectors of $\mathbf{L}_{\text{rw}}$ are cluster indicator vectors $\mathbf{1}_{A_k}$, while the eigenvectors of $\mathbf{L}_{\text{sym}}$ are additionally multiplied with $\mathbf{D}^{\frac{1}{2}}$, which might lead to undesired artifacts.

References

Ester, Martin et al. (1996). "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. KDD'96. Portland, Oregon: AAAI Press, 226231.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2009). *The Elements of Statistical Learning*. Vol. 1. Springer Series in Statistics. New York, NY, USA: Springer New York Inc.

Izenman, Alan J (Mar. 2009). *Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning*. en. Springer Science & Business Media.

Luxburg, Ulrike von (2007). "A Tutorial on Spectral Clustering". In: *Statistics and Computing* 17.4, pp. 395–416. DOI: 10.1007/s11222-007-9033-z. URL: https://doi.org/10.1007/s11222-007-9033-z.