Spectral Clustering: Intro and Mathematical Deep Dive

Gavish Bansal,Kintan Saha

Algorithm Festival, Indian Institute of Science

October 20, 2024

Introduction to Clustering

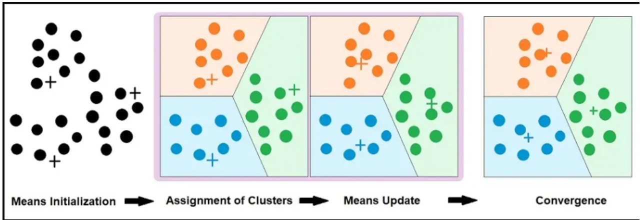
- **Clustering**: Grouping data into subsets (clusters) where data points in the same subset are similar and different sets are different.
- Traditional methods:
  - k-means, hierarchical clustering, EM clustering.
- **Challenges**: Standard clustering techniques struggle with complex data geometries because of their pre-manifested shape/distribution of data within a cluster (e.g., non-convex clusters).

Applications in real Life

- In social networks: Spectral clustering can detect communities or groups of users that are densely connected within but sparsely connected outside.
- In Bioinformatics: The graph represents genes as nodes and edges represent correlations in gene expression level. Clustering helps in identifying groups of co=expressed genes Still many more...

K-means Clustering

- **K-means**: Partition data into k clusters by minimizing the sum of squared distances between data points and cluster centroids.
- **Algorithm**:
  1. Initialize cluster centroids arbitrarily.
  2. Assign each data point to the nearest centroid.
  3. Update centroids based on the mean of data points in each cluster.
  4. Repeat steps 2-3 until convergence.



Graph Theory Basics

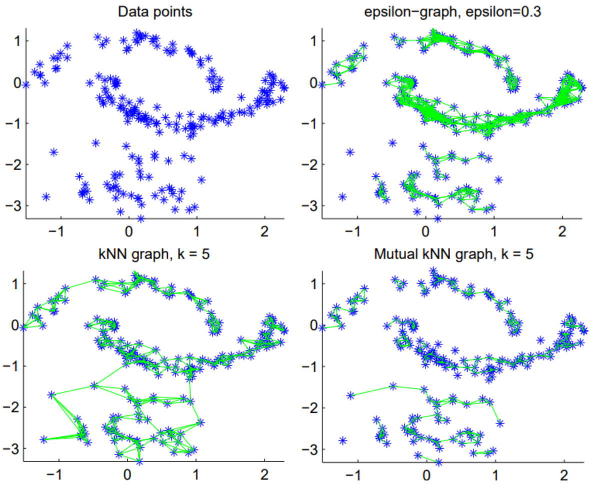
- **Graph**: A collection of nodes (vertices) connected by edges(subset of VxV).
- **Adjacency matrix**: A matrix representation of a graph where $A_{ij} = w_{ij}$ if there is an edge between nodes i and j, 0 otherwise.
- **Degree of a node**: $D_{ii} = \sum_j a_{ij}$
- **Similarity Graph**: A graph where nodes represent data points and edges represent similarities between points.

Spectral Clustering Overview

- Spectral clustering uses eigenvalue decomposition of a similarity graph derived from data.
- Graph-based approach to clustering, effectively handling complex geometries such as spirals etc.
- Steps:
  1. Construct a similarity graph.
  2. Compute the graph Laplacian.
  3. Perform spectral decomposition (eigenvalues, eigenvectors).
  4. Use eigenvectors to embed points in a lower-dimensional space.
  5. Apply k-means or similar algorithms to the embedded points.

Graph Representation of Data

- Given data points $x_1, x_2, \ldots, x_n$, construct a graph $G = (V, E)$, where:
  - Vertices V represent data points.
  - Edges E represent similarities between pairs of points.
- The similarity graph G can be constructed using:
  - ε-neighborhood graph.
  - k-nearest neighbor graph.
  - Fully connected graph (Gaussian similarity function: $W_{ij} = \exp(-\|x_i - x_j\|^2 / 2\sigma^2)$).


Data points
epsilon−graph, epsilon=0.3
kNN graph, k = 5
Mutual kNN graph, k = 5
1
0
−1
−2
−3
−1
0
1
2

Adjacency Matrix and Degree Matrix

- The **adjacency matrix** $W \in \mathbb{R}^{n \times n}$ is defined as:

$$W_{ij} = \text{similarity between points } x_i \text{ and } x_j.$$

- Gaussian similarity function: $W_{ij} = \exp(-\|x_i - x_j\|^2 / 2\sigma^2)$.
- The **degree matrix** $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix where:

$$D_{ii} = \sum_j W_{ij}.$$

Graph Laplacians

- The **unnormalized graph Laplacian** L is defined as:

$$L = D - W.$$

- **Properties of** L:
  - L is symmetric.
  - L is positive semi-definite.
  - The smallest eigenvalue $\lambda_1 = 0$ corresponds to the eigenvector $\mathbf{1}$ (the constant vector).
  - L has n non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$

Proof for Positive Semi-definiteness of L

- For any vector $f \in \mathbb{R}^n$, we have:

$$f^T L f = f^T D f - f^T W f = \sum_i D_{ii} f_i^2 - \sum_{i,j} W_{ij} f_i f_j.$$

$$f^T L f = 1/2(\sum_{i,j} W_{ij}(f_i - f_j)^2) \geq 0.$$

- Since D is diagonal and non-negative, and W is symmetric and non-negative, $f^T L f \geq 0$.
- Thus, L is positive semi-definite.

Spectral Properties of Graph Laplacians

- The eigenvalues of the Laplacian L provide important structural information about the graph.
- Let $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ be the eigenvalues of L.
- The number of connected components in the graph is equal to the multiplicity of the eigenvalue $\lambda_1 = 0$.
- Eigenvectors corresponding to small eigenvalues capture global graph structure.

Proof for Number of Connected Components

- Claim 1: A fully connected Graph has only 1 eigen value $= 0$
- Proof :

$$f^T L f = 1/2(\sum_{i,j} W_{ij}(f_i - f_j)^2) \geq 0.$$

But if f is some eigen vector with eigen value $= 0$, then $f^T L f = 0$
Now graph being connected has

$$w_{ij} \geq 0 \quad \forall (i,j) s.t\ i \neq j$$

So $f_i = f_j$ $\forall$ i,j included in eigen space of I only

Proof for Number of Connected Components

- Claim 2: A Graph with k connected components has k eigen values 0 with corresponding eigen vetors as indicator vectors.
- Proof :

$$f^T L f = 1/2(\sum_{i,j} W_{ij}(f_i - f_j)^2) \geq 0.$$

So for f to be eigen vector either $Wij = 0$ or $f_i = f_j$ $\forall$ i,j
WLOG let $P = \{i_1, i_2, .., i_{k-1}, n\}$ be the partition of V such that $\{1, 2..i_1\}, \{i_1 + 1, .., i_2\}, ..\{i_{k-1} + 1, ..n\}$ are the k connected components.
It means $f_i = f_j$ if both i,j belong to the same component and no constraint if i,j belong to different components
So Vector space spanned by Indicator vectors form eigen space with eigen value 0.

The Eigenvalue Problem

- We want to solve the eigenvalue problem:

$$Lu = \lambda u.$$

- The eigenvectors corresponding to the smallest eigenvalues describe low-frequency modes of the graph (i.e., cluster structure).
- The second smallest eigenvalue λ_2 (the *Fiedler value*) reveals the basic cluster structure of the graph.

The Fiedler Vector

- The eigenvector corresponding to the second smallest eigenvalue λ_2 is called the **Fiedler vector**.
- The Fiedler vector provides a natural cut of the graph.
- By sorting the entries of the Fiedler vector, one can divide the data points into two clusters.
- For k-way clustering, we consider the first k eigenvectors.

Why Does the Fiedler Vector Encode Structural Information?

Key Insight: The Fiedler vector is the eigenvector associated with the second smallest eigenvalue λ_2 of the Laplacian matrix $L(G)$. It encodes structural information by identifying weakly connected regions in the graph.

Spectral Partitioning and Graph Cuts:

- λ_2 is closely tied to the minimum cut of the graph.
- The Fiedler vector naturally partitions the graph by separating vertices into positive and negative values.
- Vertices with similar Fiedler vector values are likely to belong to the same strongly connected component, while vertices with large differences are weakly connected.

The Role of the Cheeger Inequality

Cheeger Inequality:

$$\frac{\lambda_2}{2} \leq h(G) \leq \sqrt{2\lambda_2}$$

- The Cheeger constant $h(G)$ measures the sparsity of the "best" graph cut, i.e., how hard it is to disconnect the graph.
- λ_2 (and thus the Fiedler vector) provides a bound on the difficulty of finding a sparse cut.
- Small λ_2: Indicates weak connectivity or bottlenecks, where only a few edges connect large portions of the graph.
- The Fiedler vector reveals these weak connections.

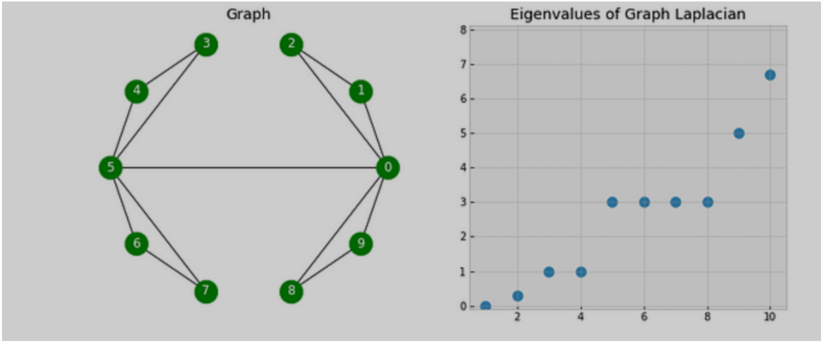
Multi-way Partitioning and Higher Eigenvalues

Generalization to Multi-way Partitioning:

- The Fiedler vector helps partition a graph into two parts. Higher eigenvectors $(\lambda_3, \lambda_4, \ldots)$ generalize this to multi-way cuts.
- Eigenvectors corresponding to λ_k help partition the graph into k parts, revealing deeper structural divisions.
- A small λ_k suggests weak connections between k regions, enabling k-way partitions.

Structural Bottlenecks:

- Small eigenvalues indicate structural bottlenecks, where only a few edges separate major components of the graph.


Graph
Eigenvalues of Graph Laplacian
0
1
2
3
4
5
6
7
8
9
10

Fiedler Cuts and the Rayleigh Quotient

Fiedler Cuts and the Rayleigh Quotient:

- The Fiedler vector minimizes the Rayleigh quotient:

$$\lambda_2 = \min_{x \perp \mathbf{1}} \frac{x^T L(G) x}{x^T x}$$

- This minimization reveals a cut in the graph that divides it at its weakest connections.
- Vertices with similar Fiedler vector values are strongly connected, while large differences in values indicate edges where the graph can be split.

Thus, the Fiedler vector encodes structural information by identifying weak points in the graph where it can be efficiently divided into components.

Graph cut point of view

- **Graph Cut**: A method to partition a graph into disjoint subsets.
- **Cut**: The total weight of edges that are removed to separate the graph into two disjoint subsets.
- **Min-Cut Problem**: Finding a cut that minimizes the total weight of the edges removed.
- **Normalized Cut (Ncut)**: A cut criterion that measures both the total weight of the edges cut and the balance of the partition.

  $$Ncut(A, B) = \frac{cut(A, B)}{|A|} + \frac{cut(A, B)}{|B|}$$

  Where ($|X|$) Number of Nodes in X.
- Spectral clustering uses the Ncut criterion to find a balanced partition of the graph.

Relaxation of the Normalized Cut Problem

- The Ncut problem is NP-hard in its discrete form.
- Spectral clustering solves a relaxed version using the eigenvalues of the graph Laplacian.
- Using given expression and taking f = indicator vector

$$f^T L f = 1/2(\sum_{i,j} W_{ij}(f_i - f_j)^2) \geq 0.$$

  we could reduce problem to:

$$\min_{y \in \mathbb{I}_{\mathbb{L}}} \frac{y^T L y}{y^T D y}, \quad \text{subject to } y^T D 1 = 0.$$

- We relax the problem to:

$$\min_{y \in \mathbb{R}^n} \frac{y^T L y}{y^T D y}, \quad \text{subject to } y^T D 1 = 0.$$

  by relaxing range of y from some Indicator vectors of graph subset to general $\mathbb{R}^n$

Relaxation Using the Rayleigh Quotient

- The problem:

$$\min_{y \in \mathbb{R}^n} \frac{y^T L y}{y^T D y}$$

is equivalent to finding the second smallest eigenvalue of the generalized eigenvalue problem:

$$Lu = \lambda Du.$$

- This is the Rayleigh quotient minimization, which spectral clustering efficiently solves.

Rayleigh Ritz Theorem

- Rayleigh Quotient $:= \frac{f^T L f}{f^T f}$
- Rayleigh Ritz Theorem states that the minimum of the Rayleigh Quotient is the smallest non zero eigen value of the matrix L.
- corresponding f is the eigen vector of the smallest non zero eigen value of L.

- Rayleigh Ritz Theorem states that the minimum of the Rayleigh Quotient is the smallest non zero eigen value of the matrix L.
- corresponding f is the eigen vector of the smallest non zero eigen value of L.

Approximate RatioCut for k = 1 with $f^T Df$

It can be shown that the ratio cut can be reduced to the following optimization problem:

$$NCut(A, A^c) = \frac{f^T Lf}{f^T f}$$

for some f such that $f^T I = 0$ and $\|f^T f\| = n$. Equivalently, we can optimize the above by approximation of the ratio cut to get an optimization problem as follows:

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

Now we can solve the above optimization problem to get the f which minimizes the Ncut.By Rayleigh Ritz theorem, the f which minimizes the above optimization problem is the Fiedler vector.Now this Fiedler vector can be used to partition the graph into two components by choosing a good threshold.

The k-Way Cut Problem

- For k-way clustering, we aim to minimize:

$$\mathrm{Ncut}(A_1, \ldots, A_k) = \sum_{i=1}^{k} \frac{\mathrm{cut}(A_i, A_i^c)}{\mathrm{vol}(A_i)}.$$

- Spectral clustering provides an efficient approximation for this multi-partition problem using the first k eigenvectors.

Important Points So Far

- Spectral clustering uses the graph Laplacian to capture the structure of the data.
- No of connected components = multiplicity of 0 eigen value in L.
- The Fiedler vector provides a natural cut of the graph dividing graphs in 2 good enough clusters.
- More generally all eigen small eigen values capture the global structure of the graph.
- Relation between optimized value of Ncut problem and λ_k.
- Relaxation of Ncut problem to Rayleigh Quotient minimization.

Developing Intuition for Spectral Clustering

- The core idea is that the first k eigenvectors contain the most significant information about the dataset (we will delve into this soon).
- The i-th row of the matrix U (as defined in the algorithm) contains the information about the i-th data point based on the first k eigenvectors.
- Once we have this matrix, we can apply the k-means algorithm to cluster the data.

Spectral Clustering Algorithm Review

- Steps:
  1. Construct a similarity graph.
  2. Compute the graph Laplacian.
  3. Perform spectral decomposition (eigenvalues, eigenvectors).
  4. Use eigenvectors to embed points in a lower-dimensional space.
  5. Apply k-means or similar algorithms to the embedded points.

Spectral Clustering Algorithm Review

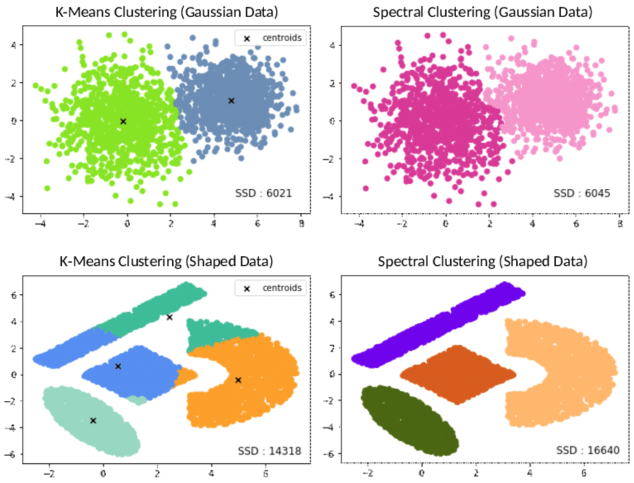
- Steps:
  1. **Construct a similarity graph:**
  2. **Compute the graph Laplacian:**
     - $$L = D - W$$
       where D is the degree matrix and W is the weight matrix.
  3. **Perform spectral decomposition:**
     - Compute the eigenvalues and eigenvectors of the Laplacian matrix L.
     - Select the first k eigenvectors corresponding to the smallest k eigenvalues. These eigenvectors will help in clustering the data.
  4. **Embed points in a lower-dimensional space:**
     - Use the selected k eigenvectors to form a matrix U, where each row corresponds to a data point.
     - Each data point is now represented in a new k-dimensional space, where points that are similar in the original space will be closer in the new space.
  5. **Apply k-means to the embedded points:**
     - Run the k-means clustering algorithm on the rows of matrix U.
     - The result is the assignment of each original data point to one of the k clusters.

Developing Intuition

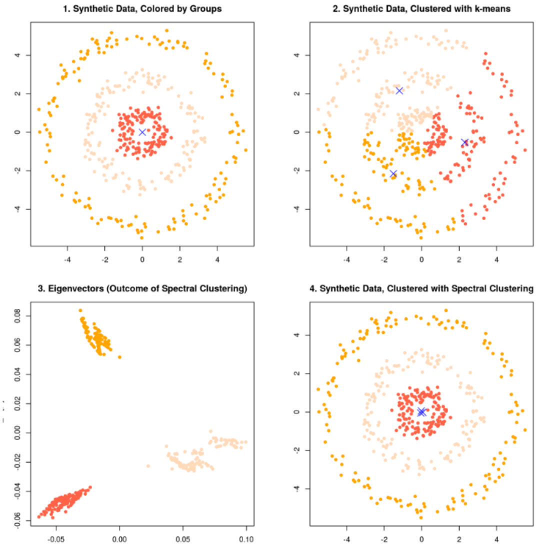
Then why not directly use the *k*-means algorithm?

- The reason is that the original dataset may not be convex.
- So, how can we be confident that the lower-dimensional representation is convex?
- In most cases, it tends to be convex after transformation.

Developing Intuition:



Developing Intuition:



Developing Intuition

Why only K vectors?

Why not (K+1)th vector?

Developing Intuition for k-Selection

- The selection of k (number of clusters) can be guided by the gap between eigenvalues.
- The difference between the k-th and $(k+1)$-th eigenvalue provides a heuristic:

$$\lambda_{k+1} - \lambda_k \gg 0 \quad \Rightarrow \quad k \text{ clusters is a good fit.}$$

- Intuitively, a large gap suggests that k eigenvectors capture most of the relevant structure in the data.
- This is because λ_k relates to the minimum value of:

$$\mathrm{Ncut}(A_1, \ldots, A_k) = \sum_{i=1}^{k} \frac{\mathrm{cut}(A_i, A_i^c)}{\mathrm{vol}(A_i)},$$

which measures the quality of the cut into k clusters.

Developing Intuition:

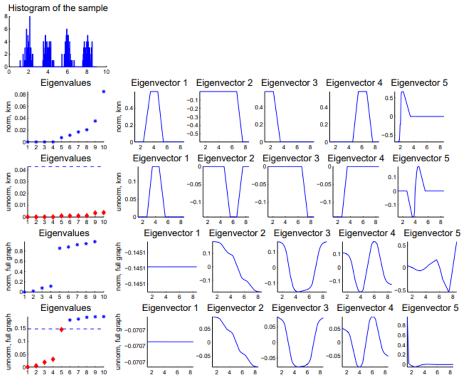


Figure 1: Toy example for spectral clustering where the data points have been drawn from a mixture of four Gaussians on $\mathbb{R}$. Left upper corner: histogram of the data. First and second row: eigenvalues and eigenvectors of L_{rw} and L based on the k-nearest neighbor graph. Third and fourth row: eigenvalues and eigenvectors of L_{rw} and L based on the fully connected graph. For all plots, we used the Gaussian kernel with $\sigma = 1$ as similarity function. See text for more details.

Eigenvalue Gap Intuition

- Suppose the $(k+1)$-th eigenvalue contains significant information that is not captured by the first k eigenvectors.
- But $\lambda_{k+1} \gg \lambda_k$, then:
  - There exists decent number of pairs (i, j) where $f_i - f_j$ is large, despite W_{ij} (the weight between them) being high.
- This indicates that the data might not be well clustered, as strong connections are being split across clusters.
- Ideally, the $(k+1)$-th eigenvector should not store significant information if k-clusters is a good model for the data.

Spectral Clustering Algorithm (Step-by-Step)

1. Construct the similarity matrix W from data.
2. Compute the degree matrix D and the Laplacian $L = D - W$.
3. Solve the eigenvalue problem $Lu = \lambda u$.
4. Select the first k eigenvectors to form matrix $U \in \mathbb{R}^{n \times k}$.
5. Use k-means clustering on the rows of U.
6. Output the clusters.

Choice of Similarity Function

- The similarity function $s(x_i, x_j)$ defines the weight of edges in the similarity graph.
- Common choices for similarity:
    - **Gaussian similarity function**:

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

    - **k-nearest neighbor similarity**.
- The choice of σ and k can significantly affect the clustering outcome.

Computational Complexity Analysis

- Computing eigenvectors for large graphs can be computationally expensive.
- For a graph with n nodes:
  - Constructing the similarity graph takes $O(n^2)$ time.
  - Eigenvalue decomposition takes $O(n^3)$ time.
- Approximations like Lanczos algorithms are used to make spectral clustering scalable.

Challenges and Limitations

- **Scalability**: Large datasets require efficient approximations for eigenvalue computation.
- **Parameter sensitivity**: The choice of k (number of clusters), similarity function, and graph construction method can greatly affect performance.
- **Noisy data**: Spectral clustering may be sensitive to noise in the graph structure.

Practical Considerations

- **Graph sparsity**: To reduce computational complexity, sparsify the similarity graph by connecting only the most similar points.
- **Dimensionality reduction**: Applying PCA or other dimensionality reduction techniques before constructing the graph can speed up computations.(Principal Component Analysis)

Comparison with Other Clustering Methods

- **k-means**:
  - Assumes convex clusters.
  - Sensitive to initialization.
- **Hierarchical clustering**:
  - No need to predefine the number of clusters.
  - Less effective for large datasets.
- **Spectral clustering**:Spectral Properties of Graph Laplacians
  - Handles non-convex clusters.
  - Computationally more expensive but more flexible.

Future Directions in Spectral Clustering

- Developing more efficient methods for large-scale spectral clustering.
- Improving robustness to noise and outliers.
- Exploring deep learning approaches combined with spectral clustering for large and complex datasets.

Conclusion and Q&A

- Spectral clustering is a powerful, flexible method for clustering data with complex structures.
- Its mathematical foundation in graph theory and eigenvalue problems makes it particularly effective for non-convex clusters.
- However, computational costs and parameter choices need careful consideration in practical applications.

Questions?