

COMPSCI 514: Algorithms for Data Science

Cameron Musco

University of Massachusetts Amherst. Fall 2023.

Lecture 19

- Problem Set 3 is due Monday at 11:59pm.

Last Class: SVD and Applications of Low-Rank Approximation

- SVD and connections to eigendecomposition and optimal low-rank approximation.
- Matrix completion
- Entity Embeddings.

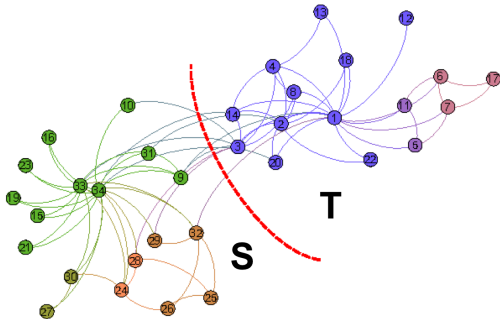
This Class: Linear Algebraic Techniques for Graph Analysis

- Start on graph clustering for community detection and non-linear clustering.
- **Spectral clustering**: finding good cuts via Laplacian eigenvectors.

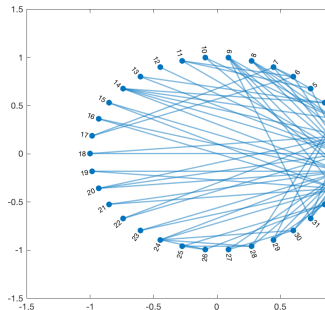
Spectral Clustering

A very common task is to **partition or cluster** vertices in a graph based on similarity/connectivity.

Community detection in naturally occurring networks.



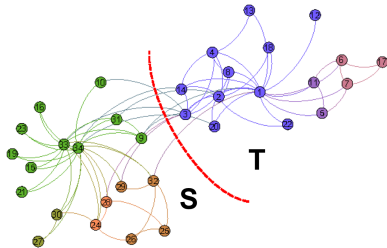
(a) Zachary Karate Club Graph



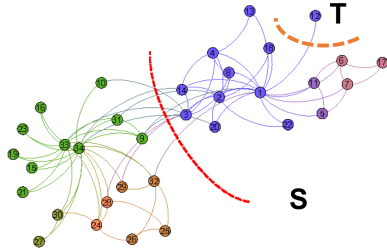
Non-linearly separable data.

Cut Minimization

Simple Idea: Partition clusters along minimum cut in graph.



(a) Zachary Karate Club Graph



(a) Zachary Karate Club Graph

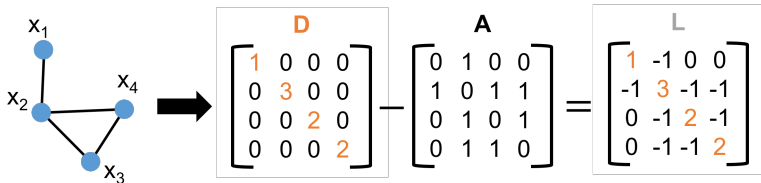
Small cuts are often not informative.

Solution: Encourage cuts that separate large sections of the graph.

- Let $\vec{v} \in \mathbb{R}^n$ be a **cut indicator**: $\vec{v}(i) = 1$ if $i \in S$. $\vec{v}(i) = -1$ if $i \in T$.
Want \vec{v} to have roughly equal numbers of 1s and -1 s. I.e.,
 $\vec{v}^T \vec{1} \approx 0$.

The Laplacian View

For a graph with adjacency matrix \mathbf{A} and degree matrix \mathbf{D} , $\mathbf{L} = \mathbf{D} - \mathbf{A}$ is the **graph Laplacian**.



For any vector \vec{v} , its 'smoothness' over the graph is given by:

$$\sum_{(i,j) \in E} (\vec{v}(i) - \vec{v}(j))^2 = \vec{v}^T \mathbf{L} \vec{v}.$$

The Laplacian View

For a cut indicator vector $\vec{v} \in \{-1, 1\}^n$ with $\vec{v}(i) = -1$ for $i \in S$ and $\vec{v}(i) = 1$ for $i \in T$:

1. $\vec{v}^T \mathbf{L} \vec{v} = \sum_{(i,j) \in E} (\vec{v}(i) - \vec{v}(j))^2 = 4 \cdot \text{cut}(S, T)$.
2. $\vec{v}^T \vec{1} = |T| - |S|$.

Want to minimize both $\vec{v}^T \mathbf{L} \vec{v}$ (cut size) and $\vec{v}^T \vec{1}$ (imbalance).

Next Step: See how this dual minimization problem is naturally solved (sort of) by eigendecomposition.

Smallest Laplacian Eigenvector

The smallest eigenvector of the Laplacian is:

$$\vec{v}_n = \frac{1}{\sqrt{n}} \cdot \vec{1} = \underset{v \in \mathbb{R}^n \text{ with } \|\vec{v}\|=1}{\text{arg min}} \quad \vec{v}^T \mathbf{L} \vec{v}$$

with eigenvalue $\lambda_n(\mathbf{L}) = \vec{v}_n^T \mathbf{L} \vec{v}_n = 0$. Why?

n : number of nodes in graph, $\mathbf{A} \in \mathbb{R}^{n \times n}$: adjacency matrix, $\mathbf{D} \in \mathbb{R}^{n \times n}$: diagonal degree matrix, $\mathbf{L} \in \mathbb{R}^{n \times n}$: Laplacian matrix $\mathbf{L} = \mathbf{A} - \mathbf{D}$.

Second Smallest Laplacian Eigenvector

By Courant-Fischer, the second smallest eigenvector is given by:

$$\vec{v}_{n-1} = \arg \min_{v \in \mathbb{R}^n \text{ with } \|\vec{v}\|=1, \vec{v}_n^T \vec{v}=0} \vec{v}^T \mathbf{L} \vec{v}.$$

If \vec{v}_{n-1} were in $\left\{-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\right\}^n$ it would have:

- $\vec{v}_{n-1}^T \mathbf{L} \vec{v}_{n-1} = \frac{4}{\sqrt{n}} \cdot \text{cut}(S, T)$ as small as possible given that $\vec{v}_{n-1}^T \vec{v}_n = \frac{1}{\sqrt{n}} \vec{v}_{n-1}^T \vec{1} = \frac{|T|-|S|}{n} = 0$.
- I.e., \vec{v}_{n-1} would indicate the smallest perfectly balanced cut.
- The eigenvector $\vec{v}_{n-1} \in \mathbb{R}^n$ is not generally binary, but still satisfies a 'relaxed' version of this property.

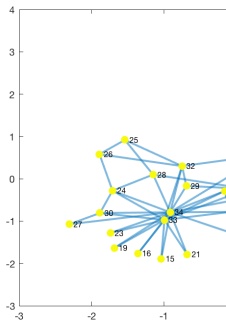
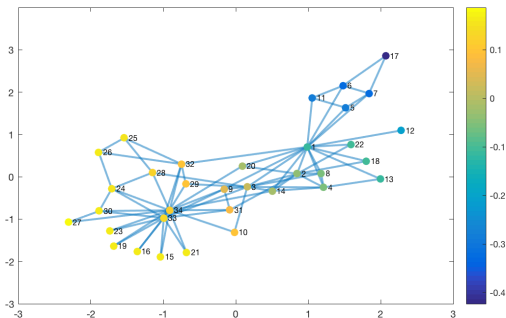
n : number of nodes in graph, $\mathbf{A} \in \mathbb{R}^{n \times n}$: adjacency matrix, $\mathbf{D} \in \mathbb{R}^{n \times n}$: diagonal degree matrix, $\mathbf{L} \in \mathbb{R}^{n \times n}$: Laplacian matrix $\mathbf{L} = \mathbf{A} - \mathbf{D}$. S, T : vertex sets on different sides of cut.

Cutting With the Second Laplacian Eigenvector

Find a good partition of the graph by computing

$$\vec{v}_{n-1} = \underset{v \in \mathbb{R}^d \text{ with } \|\vec{v}\|=1, \vec{v}^T \vec{1}=0}{\text{arg min}} \quad \vec{v}^T L \vec{v}.$$

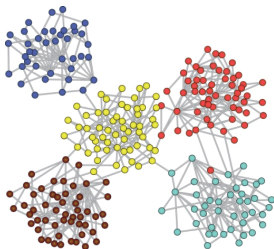
Set S to be all nodes with $\vec{v}_{n-1}(i) < 0$, T to be all with $\vec{v}_2(i) \geq 0$.



Spectral Partitioning in Practice

The Shi-Malik normalized cuts algorithm is one of the most commonly used variants of this approach, using the normalized Laplacian $\bar{\mathbf{L}} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$.

Important Consideration: What to do when we want to split the graph into more than two parts?



Spectral Clustering:

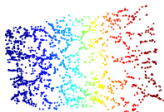
- Compute smallest k nonzero eigenvectors $\vec{v}_{n-1}, \dots, \vec{v}_{n-k}$ of $\bar{\mathbf{L}}$.
- Represent each node by its corresponding row in $\mathbf{V} \in \mathbb{R}^{n \times k}$

Laplacian Embedding

The smallest eigenvectors of $\mathbf{L} = \mathbf{D} - \mathbf{A}$ give the orthogonal 'functions' that are smoothest over the graph. I.e., minimize

$$\vec{v}^T \mathbf{L} \vec{v} = \sum_{(i,j) \in E} [\vec{v}(i) - \vec{v}(j)]^2.$$

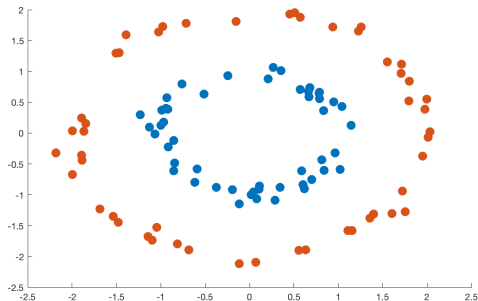
Embedding points with coordinates given by $[\vec{v}_{n-1}(j), \vec{v}_{n-2}(j), \dots, \vec{v}_{n-k}(j)]$ ensures that coordinates connected by edges have minimum total squared Euclidean distance.



- Spectral Clustering
- Laplacian Eigenmaps
- Locally linear embedding
- Isomap
- Node2Vec, DeepWalk, etc.
(variants on Laplacian)

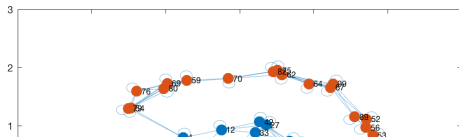
Laplacian Embedding

Original Data: (not linearly separable)



k -Nearest

Neighbors Graph:



Generative Models

So Far: Have argued that spectral clustering partitions a graph effectively, along a small cut that separates the graph into large pieces. But it is difficult to give any formal guarantee on the ‘quality’ of the partitioning in general graphs.

Common Approach: Give a natural **generative model** for random inputs and analyze how the algorithm performs on inputs drawn from this model.

- Very common in algorithm design for data analysis/machine learning (can be used to justify least squares regression, k -means clustering, PCA, etc.)
- We’ll do this next time, introducing the **Stochastic Block Model**.