Matrix Completion

X which is rank 1.

$$X = \begin{bmatrix} 1 & 2 & 3 \\ 2 & ?\ 4 & ?\ 6 \\ 3 & 6 & ?\ 9 \end{bmatrix}$$

COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Cameron Musco

University of Massachusetts Amherst. Fall 2020.

Lecture 18

LOGISTICS

- Problem Set 3 was due yesterday.
- Solutions have been posted.
- There was no quiz due this week. Will have one due next Monday as usual.

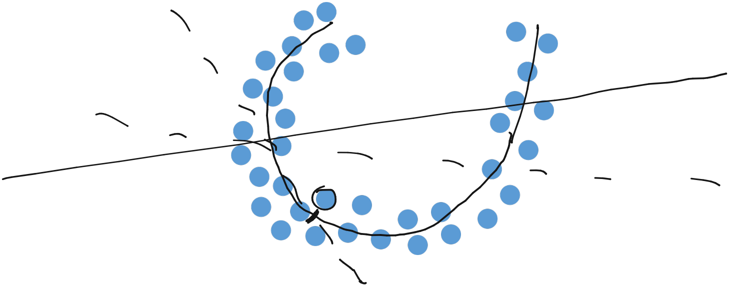
Last Class: Applications of Low-Rank Approximation

- Low-rank matrix completion (predicting missing measurements using low-rank structure).
- Entity embeddings (e.g., LSA, word embeddings). View as low-rank approximation of a similarity matrix.

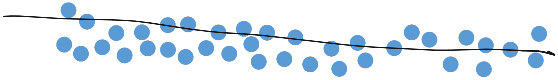
Spectral Graph Theory & Spectral Clustering.

- Low-rank approximation on graph adjacency matrix for non-linear dimensionality reduction.
- Eigendecomposition to partition graphs into clusters.
- ~~Application to the *stochastic block model* and community detection.~~

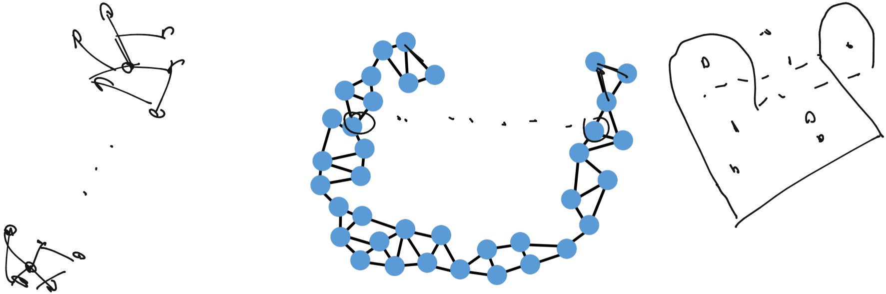
NON-LINEAR DIMENSIONALITY REDUCTION



Is this set of points compressible? Does it lie close to a low-dimensional subspace? (A 1-dimensional subspace of $\mathbb{R}^d$.)



Is this set of points compressible? Does it lie close to a low-dimensional subspace? (A 1-dimensional subspace of $\mathbb{R}^d$.)



Is this set of points compressible? Does it lie close to a low-dimensional subspace? (A 1-dimensional subspace of $\mathbb{R}^d$.)

A common way of automatically identifying this non-linear structure is to connect data points in a graph. E.g., a k-nearest neighbor graph.

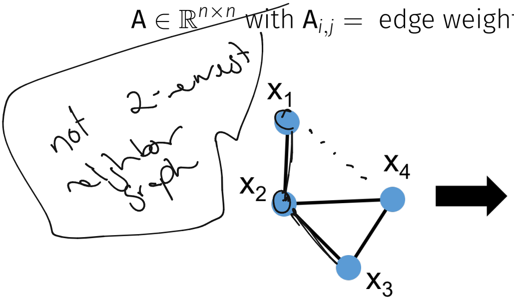
- Connect items to similar items, possibly with higher weight edges when they are more similar.

Once we have connected n data points $x_1, \ldots, x_n$ into a graph, we can represent that graph by its (weighted) adjacency matrix.

$$\mathbf{A} \in \mathbb{R}^{n \times n} \text{ with } \mathbf{A}_{i,j} = \text{ edge weight between nodes } i \text{ and } j$$

Once we have connected n data points $x_1, \ldots, x_n$ into a graph, we can represent that graph by its (weighted) adjacency matrix.

$$\mathbf{A} \in \mathbb{R}^{n \times n} \text{ with } \mathbf{A}_{i,j} = \text{ edge weight between nodes } i \text{ and } j$$

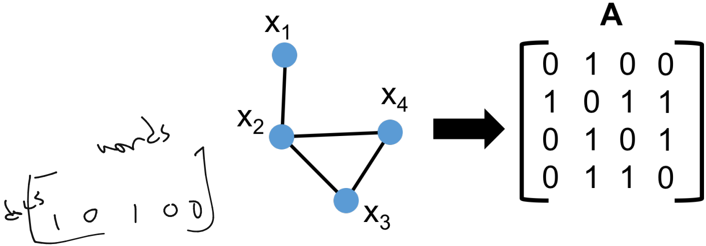


$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

LINEAR ALGEBRAIC REPRESENTATION OF A GRAPH

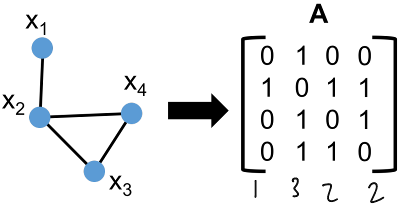
Once we have connected n data points $x_1, \ldots, x_n$ into a graph, we can represent that graph by its (weighted) adjacency matrix.

$$\mathbf{A} \in \mathbb{R}^{n\times n} \text{ with } \mathbf{A}_{i,j} = \text{ edge weight between nodes } i \text{ and } j$$



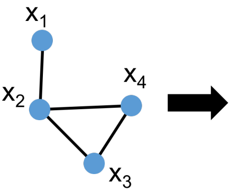
In LSA example, when $\mathbf{X}$ is the term-document matrix, $\mathbf{X}^T\mathbf{X}$ is like an adjacency matrix, where $word_a$ and $word_b$ are connected if they appear in at least 1 document together (edge weight is # documents they appear in together).

NORMALIZED ADJACENCY MATRIX



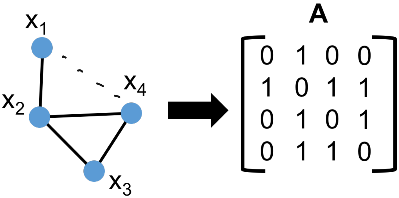
What is the sum of entries in the i^{th} column of A?

NORMALIZED ADJACENCY MATRIX



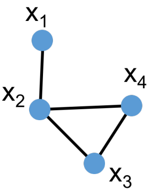
$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

What is the sum of entries in the i^{th} column of A? The (weighted) degree of vertex i.



What is the sum of entries in the i^{th} column of A? The (weighted) degree of vertex i.

Often, $\mathbf{A}$ is normalized as $\bar{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ where $\mathbf{D}$ is the degree matrix.



$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

1 3 2 2

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}$$

What is the sum of entries in the i^{th} column of A? The (weighted) degree of vertex i.

Often, $\mathbf{A}$ is normalized as $\bar{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ where $\mathbf{D}$ is the degree matrix.

NORMALIZED ADJACENCY MATRIX

symmetric

AD^{-1} $D^{-1}A$

$$\bar{\mathbf{A}} = \mathbf{D}^{-1/2}\,\mathbf{A}\,\mathbf{D}^{-1/2}$$

$$\begin{bmatrix} 0 & \frac{1}{\sqrt{3}} & 0 & 0 \\ \frac{1}{\sqrt{3}} & 0 & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} \\ 0 & \frac{1}{\sqrt{6}} & 0 & \frac{1}{2} \\ 0 & \frac{1}{\sqrt{6}} & \frac{1}{2} & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix}$$

What is the sum of entries in the i^{th} column of A? The (weighted) degree of vertex i.

Often, $\mathbf{A}$ is normalized as $\bar{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ where $\mathbf{D}$ is the degree matrix.

$$\underbrace{\begin{bmatrix} 0 & \frac{1}{\sqrt{3}} & 0 & 0 \\ \frac{1}{\sqrt{3}} & 0 & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} \\ 0 & \frac{1}{\sqrt{6}} & 0 & \frac{1}{2} \\ 0 & \frac{1}{\sqrt{3}} & \frac{1}{2} & 0 \end{bmatrix}}_{\bar{\mathbf{A}}} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix}}_{\mathbf{D}^{-1/2}} \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix}}_{\mathbf{D}^{-1/2}}$$

What is the sum of entries in the i^{th} column of A? The (weighted) degree of vertex i.

Often, $\mathbf{A}$ is normalized as $\bar{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ where $\mathbf{D}$ is the degree matrix.

Spectral graph theory is the field of representing graphs as matrices and applying linear algebraic techniques.

ADJACENCY MATRIX EIGENVECTORS

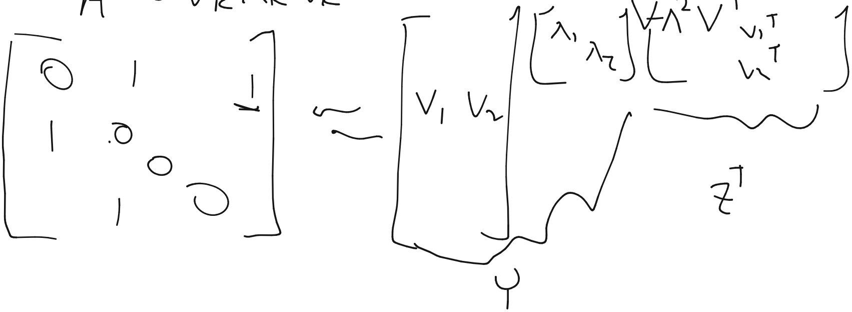
$(X^TX)_{ab}$ = # docs that word a & word b appear in

How do we compute an optimal low-rank approximation of **A**?

- Project onto the top k eigenvectors of $\mathbf{A}^T\mathbf{A} = \mathbf{A}^2$. These are just the eigenvectors of **A**.

$\mathbf{A} = V\Lambda V^T$, $\quad \mathbf{A}^2 = V\Lambda V^T V\Lambda V^T = V\Lambda^2 V^T$

$A \approx V_k \Lambda_k V_k^T$



$\langle y(i), z(j)\rangle \approx 1$

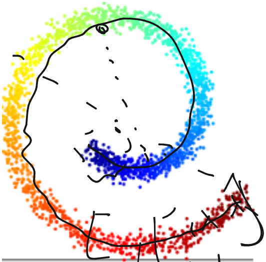
$\langle y(i), z(j)\rangle \approx 0$

How do we compute an optimal low-rank approximation of $\mathbf{A}$?

- Project onto the top k eigenvectors of $\mathbf{A}^T\mathbf{A} = \mathbf{A}^2$. These are just the eigenvectors of $\mathbf{A}$.

- Similar vertices (close with regards to graph proximity) should have similar embeddings.

original data

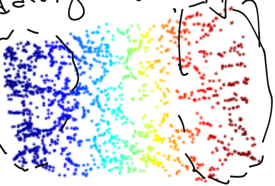


① - produce a k nearest neighbor graph

- connect points close together along spiral

② - find Y, Z s.t. $A \approx Y Z^T$

embedding



Y(i,:) ~

rows Y

Y(:, 2)

X k dimensional data point

↓

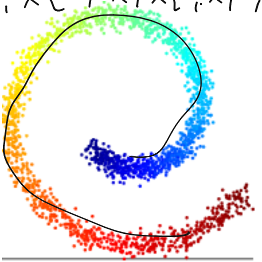
graph

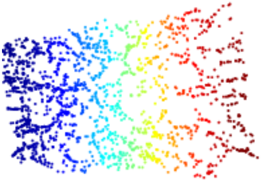
↓

embedding

(x_1, x_2)

$(x_1, x_2, x_1 x_2, x_1^2, x_2^2)$





What other methods do you know for embedding or representing data points with non-linear structure?

→ kernel PCA

Kernel methods

random walks on graphs

feature transformations

neural networks

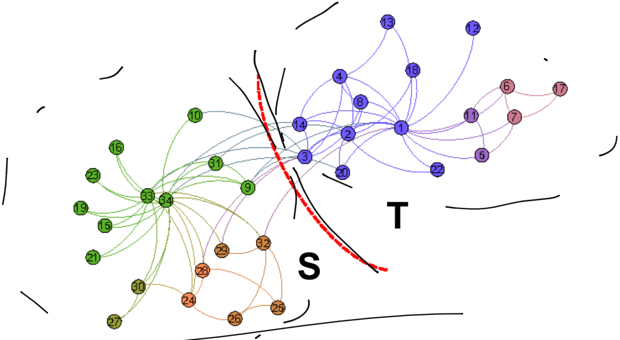
DeepWalk

node2Vec

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

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

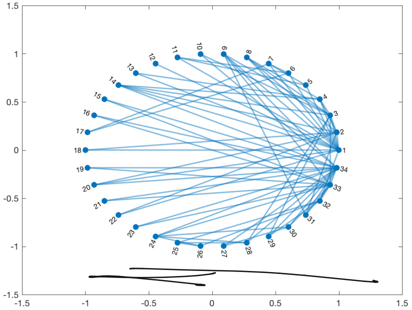
Community detection in naturally occurring networks.



(a) Zachary Karate Club Graph

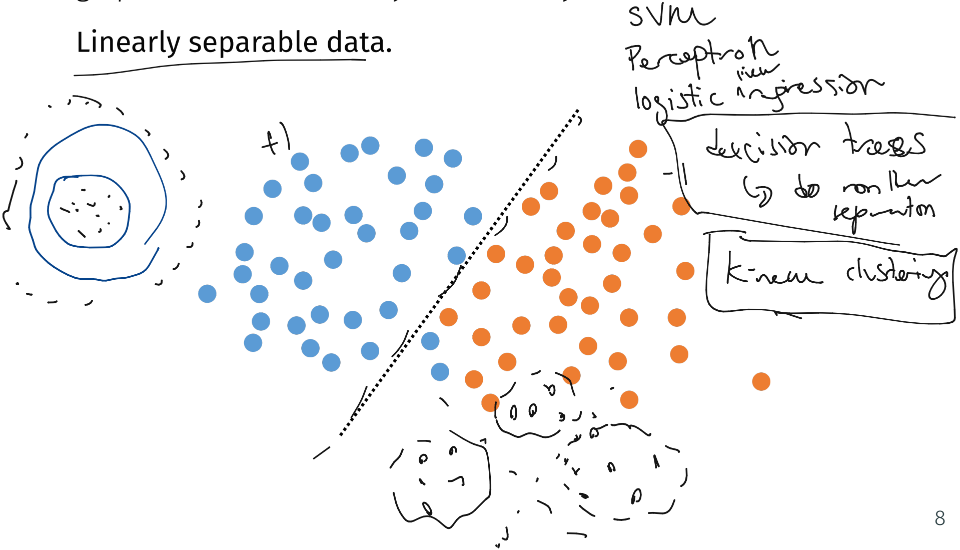
A very common task is to partition or cluster vertices in a graph based on similarity/connectivity.

Community detection in naturally occurring networks.



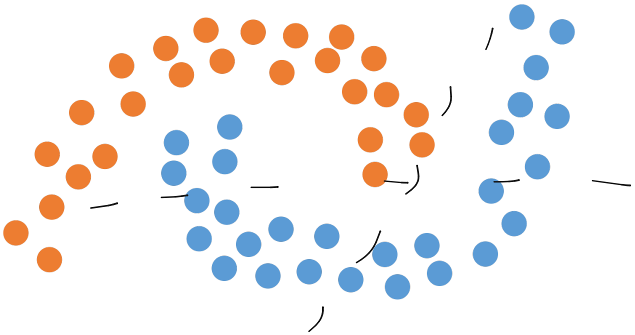
A very common task is to partition or cluster vertices in a graph based on similarity/connectivity.

Linearly separable data.



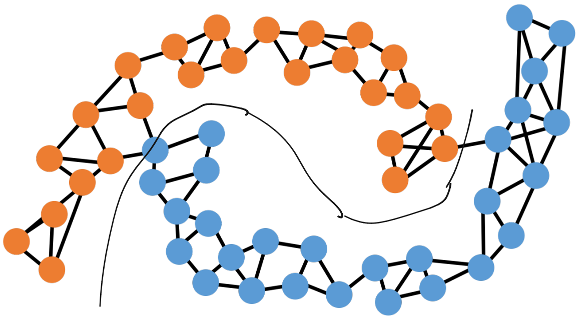
A very common task is to partition or cluster vertices in a graph based on similarity/connectivity.

Non-linearly separable data k-nearest neighbor graph.



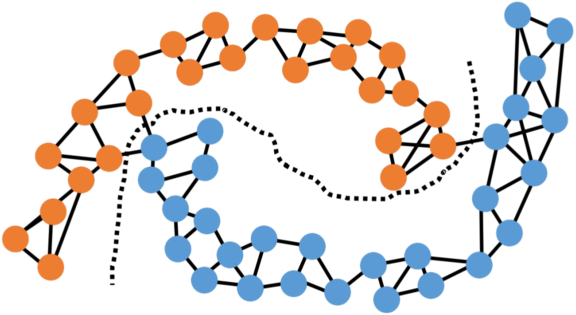
A very common task is to partition or cluster vertices in a graph based on similarity/connectivity.

Non-linearly separable data k-nearest neighbor graph.



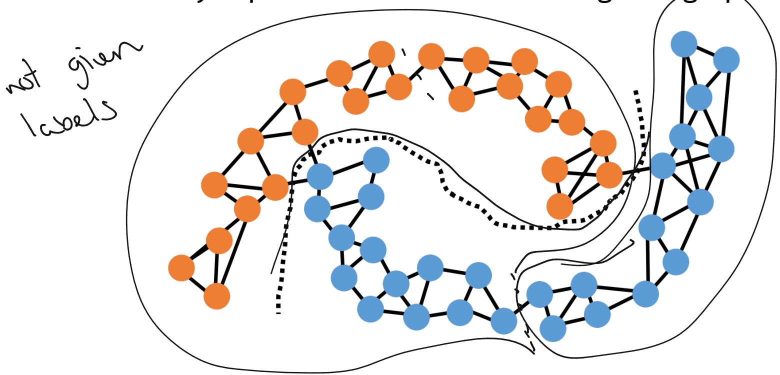
A very common task is to partition or cluster vertices in a graph based on similarity/connectivity.

Non-linearly separable data k-nearest neighbor graph.



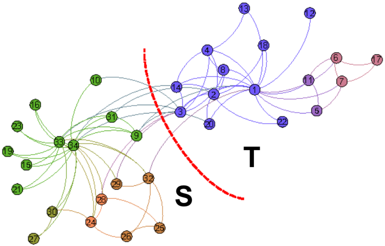
A very common task is to partition or cluster vertices in a graph based on similarity/connectivity.

Non-linearly separable data k-nearest neighbor graph.



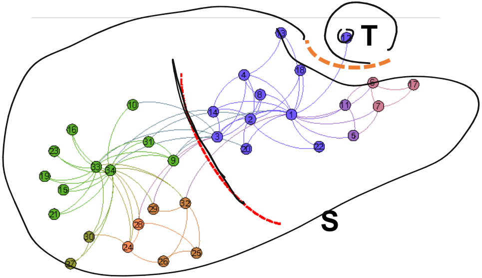
This Class: Find this cut using eigendecomposition. First – motivate why this type of approach makes sense.

Simple Idea: Partition clusters along minimum cut in graph.



(a) Zachary Karate Club Graph

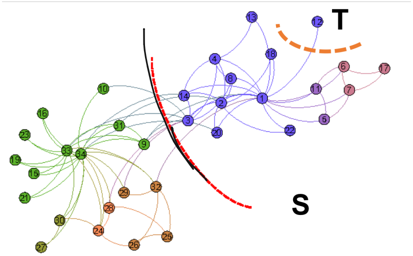
Simple Idea: Partition clusters along minimum cut in graph.



(a) Zachary Karate Club Graph

Small cuts are often not informative.

Simple Idea: Partition clusters along minimum cut in graph.

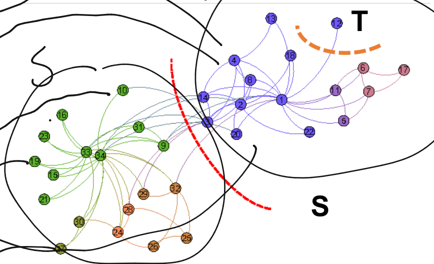


(a) Zachary Karate Club Graph

Small cuts are often not informative.

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

Simple Idea: Partition clusters along minimum cut in graph.



(a) Zachary Karate Club Graph

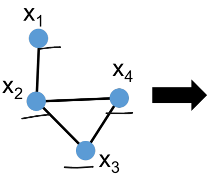
Small cuts are often not informative.

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 -1s. I.e., $\vec{v}^T\vec{1} \approx 0$.

THE LAPLACIAN VIEW

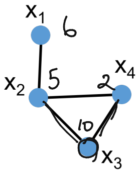
For a graph with adjacency matrix **A** and degree matrix **D**, $\mathbf{L} = \mathbf{D} - \mathbf{A}$ is the graph Laplacian.



$$\overset{\mathbf{D}}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}} - \overset{\mathbf{A}}{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}} = \overset{\mathbf{L}}{\begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 3 & -1 & -1 \\ 0 & -1 & 2 & -1 \\ 0 & -1 & -1 & 2 \end{bmatrix}}$$

$[6\ 5\ 10\ 2]^T [\ L\ ] [6\ 5\ 10\ 2]^T = \sum_{i,j \in E} (v_i - v_j)^2$

For a graph with adjacency matrix **A** and degree matrix **D**, **L** = **D** − **A** is the graph Laplacian.



$$\overset{\mathbf{D}}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}} - \overset{\mathbf{A}}{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}} = \overset{\mathbf{L}}{\begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 3 & -1 & -1 \\ 0 & -1 & 2 & -1 \\ 0 & -1 & -1 & 2 \end{bmatrix}}$$

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

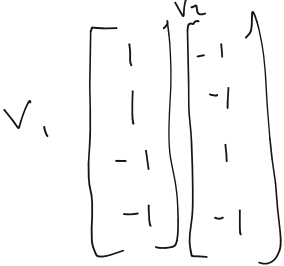
$$V = \begin{bmatrix} 6 \\ 5 \\ 10 \\ 2 \end{bmatrix} \qquad \sum_{(i,j)\in E} (\vec{v}(i) - \vec{v}(j))^2 = \vec{v}^T \mathbf{L} \vec{v}.$$

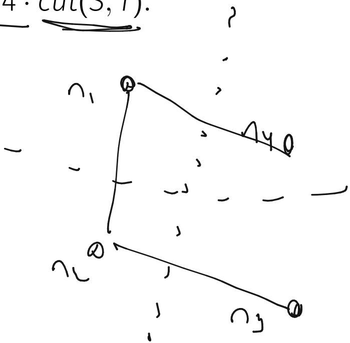
Exercise: prove this.

$(6-5)^2 + (5-2)^2 + (5-10)^2 + (10-2)^2$

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





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

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

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

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

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 by eigendecomposition.