SPECTRAL CLUSTERING via SPECTRAL GRAPH THEORY

GRAPH $G = (V, E)$

Several matrices associated to G:
- adjacency matrix
- Vertices vs. edges matrix
-

adjacency matrix

$A = V \begin{bmatrix} \\ a_{ij} \\ \\ \end{bmatrix}$ (columns: V)

$$a_{ij} = \begin{cases} 1, & (i,j) \in E \\ 0, & (i,j) \notin E \end{cases}$$

"Matrices ≡ Graphs"
A one-to-one with G.

Linear Algebra for Matrices → the mathematical foundations of linear Algebra transferred to graph!!

Spectral Graph Theorem ≠

Linear Algebra of Matrices ≡ continuous Math
Eigenvectors
Eigen values
} ⟺ help with NP hard pbs

Graph Theory ≡ discrete math

SPECTRAL CLUSTERING

- A very practical method
- clustering not a well defined PB.

Two axioms:

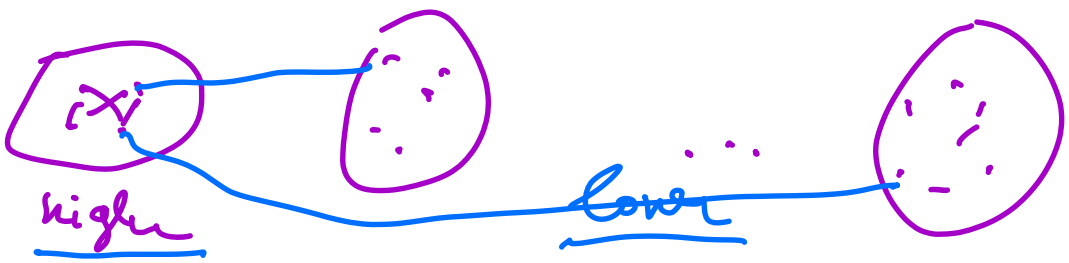
Clustering is a partition of the input S into classes $C_1, C_2, \ldots, C_k$ such that the following properties are satisfied:

① There is a similarity measure

s_{ij} = the similarity between x_i and x_j

Input: $x_1, x_2, \ldots, x_n$ $\quad 1 \le i, j \le n$, $i \neq j$

① In each class C_i the the pairwise similarities between the elements in C_i are higher than similarities between in-C_i vs. outside-C_i

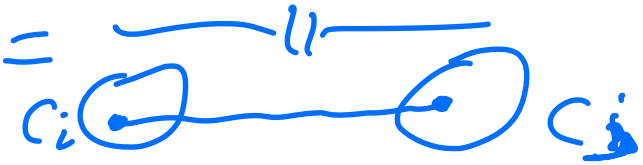


higher inside C_i, $\forall i$

lower inbetween C_i, C_j.

- not a unique way to measure "inside" or "inbetween"

inside ≡ sum of pairs of similarities

in-between = 

other choices: the min inside similarity to be larger than min best in between

⋮

INPUT:

Given a set of points $x_1, \ldots, x_n$ and a similarity s_{ij} $\forall i, j$, $i \neq j$.

GOAL/COMPUTE:

a partition of the input set of points such that points in the same clap are similar, and points in different groups are

dissimilar to each other.

Def Similarity Graph $G=(V,E)$

vertex v_i represents point x_i

Edge e_{ij} between v_i and v_j is labeled with weight s_{ij}, if $s_{ij}>0$ (another def if $s_{ij}>\theta$ = threshold)

Graph Theory notation

$G=(V,E)$, $V=\{v_1,\dots,v_n\}$

each edge has a weight

$w(v_i,v_j)=w_{ij}\geq 0$

The weighted adjacency matrix

$$W = (w_{ij})_{\substack{i=1,\dots,n \\ j=1,\dots,n}}$$

If $w_{ij} = 0$ the vertices are not connected.

Can undirected graph

$$w_{ij} = w_{ji}$$ symmetric

The degree of a vertex $v_i \in V$ is

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

the sum is over weights that are $\neq 0$.

The Degree Matrix D is defined as the diagonal matrix with degrees $d_1, d_2, \ldots, d_n$ on the diagonal

$$D = \begin{bmatrix} d_1 & & 0 \\ & d_2 & \\ 0 & & \ddots \\ & & & d_n \end{bmatrix}$$

- Given a set of vertices $A \subset V$ we denote its complement by $\bar{A} = V \setminus A$.

We define the indicator vector $\mathbf{1}_A = (f_1, f_2, \ldots, f_n) \in \mathbb{R}^n$

$f_i \in \{0, 1\}$

$\mathbb{R}^n$ the n-dimensional real vector space

with entries

$$f_i = \begin{cases} 1, & \text{if } v_i \in A \\ 0, & \text{otherwise} \end{cases}$$

- We write for $i \in A$ as a short for $\equiv \{i \mid v_i \in A\}$
- To measure the "size" of a set $A \subset V$ we have two notions:

usual: $|A|$ = the number of elements in A

new: $\mathrm{Vol}(A) = \sum_{i \in A} d_i$

The main conception in Spectral clustering:

The Graph Laplacian

- Laplacian Matrices
- Assume we have a graph $G = (V, E)$ undirected

graph weight matrix is

$$W = (w_{ij}) \quad \begin{matrix} i = 1, \dots, n \\ j = 1, \dots, n \end{matrix}$$

$w_{ij} = w_{ji} \geq 0$

W an important matrix

our def of G.

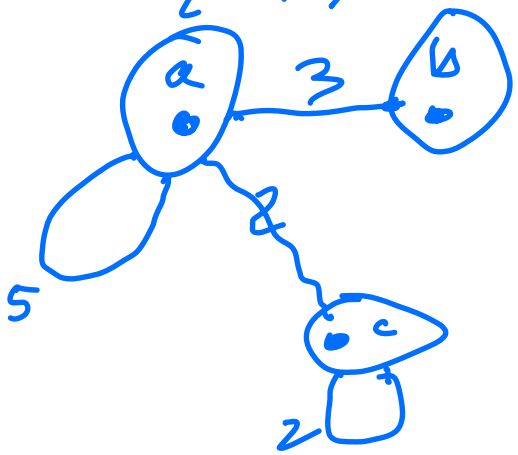
has all info from G

example $W = \begin{bmatrix} 5, 3, 2 \\ 3, 0, 0 \\ 2, 0, 2 \end{bmatrix}$

what is the graph G_W defined by W:

$G_W = (V, E)$

$V = \{a, b, c\}$



$W = G_W$

- Eigenvectors of a matrix are not normalized to norm 1:

= $\mathbb{1}$ constant vector all 1's

$a\mathbb{1}$, $a \neq 0$ we have

$\mathbb{1}$ and $a\mathbb{1}$ considered as the same vector same class

- For a matrix consider the eigenvalues of the matrix

$$Ax = \lambda x$$

↑ evector

, λ is an eigenvalue

- We consider the eigenvalues of a matrix A increasingly respecting their multiplicities.
- Our algorithms have this form: (PCA like dimensionality reduction)

We look at at the first k eigenvectors that correspond to the first k eigenvalues (the smallest k eigenvalues)

$$\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$$

the eigenvalues ordered increasingly.

The First Graph Laplacian

"The UNNORMALIZED GRAPH LAPLACIAN"

$$L = D - W$$

this is the def of unnormalized graph Laplacian.

D = diagonal matrix of G

W = weight matrix of G

Theorem

The matrix L satisfies:

① For every vector $f \in \mathbb{R}^n$ we have:

$f_1, f_2, \ldots, f_n$

$v_1, v_2, \ldots, v_n$

f^T or f' Transpose

$$f'Lf = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2$$

f' = the transpose of f

(2) L is symmetric and positive definite.

(3) The smallest eigenvalue of L is $=0$ and the corresponding eigenvector for 0 is all 1's

$$0 = \lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \cdots \leq \lambda_n$$

λ_2 = second smallest eigenvalue

(4) L has n non-negative real value eigenvalues.

Another theorem

how linear algebra concepts of the adjacency matrix of G can compute for us graph properties.

THEOREM (Number of connected components of a graph)

Let G be an undirected graph with $W_{ij} \geq 0$ weights. Then the multiplicity k of the eigenvalue 0 of L equals the number of connected components of G: $A_1, A_2, \ldots, A_k$

UNNORMALIZED SPECTRAL CLUSTERING Algorithm

INPUT: . Similarity matrix $S \in \mathbb{R}^{n \times n}$

$x_1, x_2, \ldots, x_n$ clusters

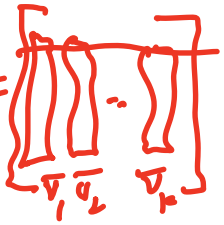
S_{ij} = some similarity between x_i and x_j

. k the number of clusters

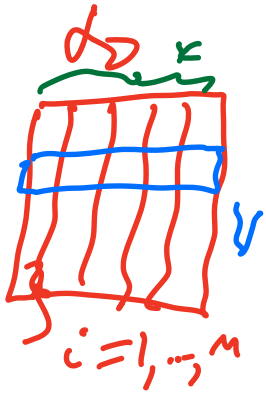
- let W be its weight adjacency matrix
- Compute the unnormalized Laplacian $L = D - W$
- Compute $V \in \mathbb{R}^{n \times k}$ be the matrix containing the first k-eigenvectors of L

$\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_k$

V contains $\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_k$ as columns



- For $i = 1, \dots, n$ let $\bar{y}_i \in \mathbb{R}^k$ be the vector corresponding the i^{th} row of V



- Cluster the points $\{\bar{y}_i\}_{i=1,\dots,n}$ with the k-means clustering algorithm into clusters

$$C_1, C_2, \dots, C_k$$

OUTPUT: clusters $A_1, A_2, \dots, A_k$ with

$$A_i = \{ j \mid \bar{y}_j \in C_i \}$$

$1 \le i \le n$

Next:

TWO INTUITIONS

- Cuts in graphs
- Random walks on graphs

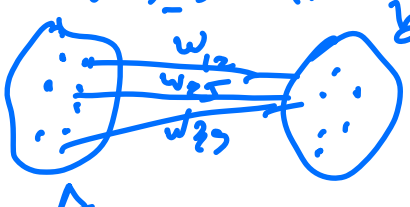
Obj. functions

$$\text{RatioCut}(A_1, \ldots, A_k) = \sum_{i=1}^{k} \frac{\text{Cut}(A_i, \bar{A}_i)}{|A_i|}$$

MIN

$\bar{A}$ = complement of A

$\text{Cut}(A, \bar{A})$ = sum of weights of edges between A and $\bar{A}$



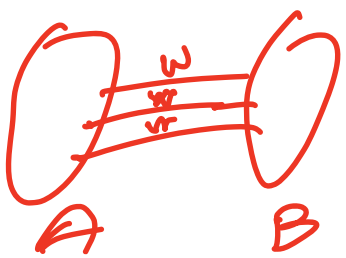
$A \cup \bar{A} = V$

$A \cap \bar{A} = \emptyset$

$$\text{NCut}(A_1, \ldots, A_k) = \sum_{i=1}^{k} \frac{\text{cut}(A_i, \bar{A}_i)}{\text{vol}(A_i)}$$

GRAPH CUT INTUITION

$$\text{Cut}(A, B) = \sum_{\substack{i \in A \\ j \in B}} w_{ij}$$



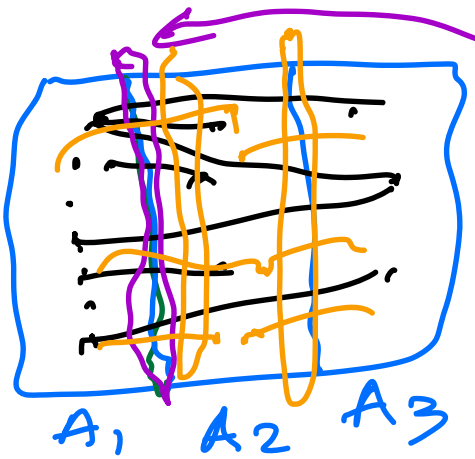
- Given a similarity graph with W as adjacency matrix, the simplest way to think about partitioning G is to consider a MINCUT Problem:

  choose a partition $A_1, A_2, \ldots A_k$

which minimizes

$$Cut(A_1, \ldots, A_k) = \sum_{i=1} Cut(A_i, \bar{A}_i)$$

$\bar{A}_i$ = Complement of A_i



Minimize

$Cut(A_1, \bar{A}_1)$
+
$Cut(A_2, \bar{A}_2)$
+
$Cut(A_3, \bar{A}_3)$

we also want partitions where all the clusters are "reasonable large".

Balanced cut: NP-complete

relaxation: real values

"Unbalanced" — or -par can be solved exactly.

The exact solution for $k=2$ of RatioCut

MIN RatioCut $(A, \bar{A})$
$A \subset V$

We are going to use Linear Algebra re-formulation of the objective function:

Consider $f \in \mathbb{R}^n$ $f = (f_1, \ldots, f_n)$

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

$$f^T L f = \sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2$$

$$= \cdots$$

$$= 2 \cdot |V| \cdot \text{RatioCut}(A, \bar{A})$$

Min $f^T L f$ is equivalent to minimizing RatioCut$(A, \bar{A})$

Note. $\sum_{i=1}^{n} f_i = 0$

f is orthogonal to the $\mathbb{1}$ vector

- $\|f\|_F^2 = n$

So minimizing RatioCut$(A, \bar{A})$ is equivalent

$$\min_{f \in \mathbb{R}^n} f^T L f \quad \text{subject to}$$

f_i = binary

$$f \perp \mathbb{1}$$

$$\|f\| = \sqrt{n}$$

This is NP-complete so solve exactly: every entry of f is binary.

Relaxation: Make the elem. of f real numbers $f_i \in \mathbb{R}$

The Relaxed optimization is

$$\min_{f \in \mathbb{R}^n} f^T L f, \quad \text{subject to} \quad f \perp \mathbb{1}, \quad \|f\| = \sqrt{n}$$

This optimization can be solved exactly by using λ_2 of L (assume G connected)

$$\lambda_0 = 0 < \lambda_1 \leq \dots$$

λ_1: smallest non-zero eigenvalue of L

We obtain an approximation of RatioCut by the second eigenvalue of L.

Now obtain the partition of G: MinCut.

We can use the sign of f_i to obtain the partition.

$$\begin{cases} v_i \in A, & \text{if } f_i \geq 0 \\ v_i \in \bar{A}, & \text{if } f_i < 0 \end{cases}$$

It is easy for $k=2$.

When $k>2$:

Spectral clustering construct

$C, \bar{C}$

$$\begin{cases} v_i \in A, & \text{if } f_i \in C \\ v_i \in \bar{A}, & \text{if } f_i \in \bar{C} \end{cases}$$

C is a partition of the rows

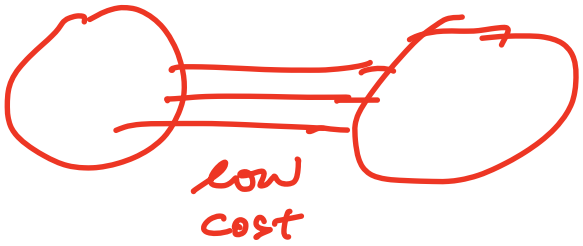
The Normalized Laplacians

$$L_{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$$

symmetric

$$L_{rw} = D^{-1}L = I - D^{-1}W$$ | rw random walk

Random walk = Spends most time in the clusters and rarely jumps between



{ Cut Random walk }

---

rw matrix

$P = \{p_{ij}\}$

Random walk matrix P

$p_{ij} = \frac{w_{ij}}{d_i}$ = prob of moving from v_i to v_j

$P = D^{-1}W$

Normalized Spectral Clustering

Shi & Malik (2000)

INPUT: • similarity matrix $S \in \mathbb{R}^{n \times n}$

• k = the number of clusters

• Construct similarity graph G

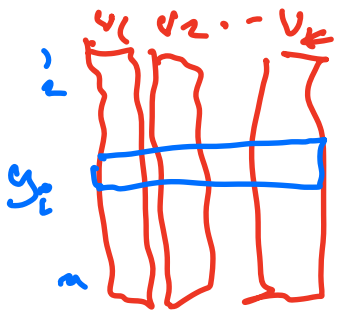
• Let W its weighted adjacency matrix

• Compute L the unnormalized Laplacian

⊙ Compute the first k eigenvectors $v_1, v_2, \ldots, v_k$ of the GENERALIZED eigenvalue problem:

$$Lv = \lambda Dv$$

- Compute $V \in \mathbb{R}^{n \times k}$ to be the matrix containing the vectors $v_1, v_2, \ldots, v_k$ as columns
- For $i = 1, \ldots, m$, Let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i^{th} row of V



- Cluster the points $\{y_i\}_{i=1}^{m}$ in $\mathbb{R}^k$ with the k-means algorithm into clusters: $C_1, C_2, \ldots, C_k$.

OUTPUT: Clusters

$A_1, A_2, \ldots, A_k$:

where $A_i = \{j \mid y_j \in C_i\}$