6.207/14.15: Networks
Lecture 8: Clustering
Outline

Spectral clustering
   Graph Laplacian
   Eigenvector for clustering

Modularity maximization
   Deciding number of clusters
   Spectral clustering with a variation

References:
   Newman, Chapter 6 (6.13)
   Newman, Chapter 11 (11.5, 11.8)
Example 1: Polarization in Content Networks

Figure: The network structure of political blogs prior to the 2004 U.S. Presidential election reveals two natural and well-separated clusters (Adamic and Glance, 2005)
Example 2: Groups in Social Networks

Figure: The social network of friendships within a 34-person karate club provides clues to the fault lines that eventually split the club apart (Zachary, 1977)

Clustering

What is clustering (in the context of network)?

Dividing nodes into disjoint groups (clusters)
In a “sensible” manner

What is “sensible”? 

There is no one definition. Context dependent.
The reason why there are many different algorithms.
In our examples, the outcome seems “sensible”

Evaluating clustering algorithm

For dataset where ground truth (clustering) is known
Can evaluate outcome of algorithm using various score (e.g. F1-score)
Why clustering?

Useful tool for exploratory analysis of network data, to begin with

Captures (hidden) social structure in social network, e.g.

- people with similar “opinions”
- people who are “friends”
- people with similar “preferences” or “tastes”

→ recommendation system

More generally, identifies “heterogenous” components in network

Different modes of population preferences

In a probabilistic model

it captures “mixture” component of the distribution
Graph partition

We shall start with simplest clustering problem: Graph partition

Given (undirected) graph $G$ over $n$ nodes $N$ and edges $E$
And given numbers $1 \leq n_1, n_2 < n$ such that $n_1 + n_2 = n$
Partition $N$ into two disjoint sets of sizes $n_1$ and $n_2$
So as to minimize number of edges between partitions

Formally:

for each node $i \in N$, assign $s_i \in \{-1, 1\}$
the objective that needs to be minimized is

$$C(s) = \sum_{i: s_i = +1} \sum_{j: s_j = -1} A_{ij}$$

where $A = [A_{ij}]$ is the adjacency matrix; $s = (s_i)$
Graph partition

A trick: for any $s_i, s_j \in \{-1, 1\}$:

$$\frac{1}{2} \left( 1 - s_i s_j \right) = \begin{cases} 1 & \text{if } s_i \neq s_j \\ 0 & \text{if } s_i = s_j. \end{cases}$$

Therefore

$$C(s) = \sum_{i: s_i = +1, j: s_j = -1} A_{ij}$$

$$= \frac{1}{2} \sum_{i,j} \left( 1 - s_i s_j \right) A_{ij}$$

$$= \frac{1}{2} \left( \sum_{i,j} A_{ij} - \sum_{i,j} s_i A_{ij} s_j \right).$$
Graph partition

Observe

$$\sum_{i,j} A_{ij} = \sum_i (\sum_j A_{ij}) = \sum_i k_i$$

$$= \sum_i (\sum_j D_{ij}) = \sum_i D_{ij}$$

$$= \sum s_i D_{ij} s_j = s^T D s,$$

where $D = \text{diag}(k_i)$ is diagonal matrix; $s_i^2 = 1$ for all $s \in \{-1, 1\}^n$.

$$C(s) = \frac{1}{2} \left( s^T D s - s^T A s \right) = \frac{1}{2} s^T (D - A) s.$$
Graph Laplacian

Graph Laplacian matrix $L = D - A$. That is,

$$C(s) = \frac{1}{2} s^T L s.$$ 

Another view of $C(s)$:

$$C(s) = \sum_{i:s_i = +1 \atop j:s_j = -1} A_{ij}$$

$$= \frac{1}{4} \sum_{i,j} (s_i - s_j)^2 A_{ij}.$$ 

More generally, for any $s \in \mathbb{R}^n$

$$s^T L s = \frac{1}{2} \sum_{i,j} (s_i - s_j)^2 A_{ij} \geq 0.$$
Graph Laplacian

Revised (*relaxed*) goal:

\[
\text{maximize } s^T L s \text{ for any } s \in \mathbb{R}^n \\
\text{so that } \sum_i s_i^2 = n \text{ and } \sum_i s = n_1 - n_2
\]

Lagrangian associated with this optimization

\[
g(s, \lambda, \mu) = s^T L s + \lambda (n - s^T s) + \mu (n_1 - n_2 - 1^T s)
\]

Taking partial derivative

\[
\frac{\partial g}{\partial s_j} = 2 \sum_j L_{ij} s_j - 2 \lambda s_i - \mu
\]
Graph Laplacian

Setting partial derivate to 0 for optimal assignment

$$Ls = \lambda s + \frac{\mu}{2} 1$$

Since $1^T L = 0$ and $1^T s = n_1 - n_2$, we have

$$0 = 1^T L s = \lambda 1^T s + \frac{\mu}{2} 1^T 1$$

That is

$$\frac{\mu}{2} = -\lambda \frac{n_1 - n_2}{n}$$

Therefore, optimal value $s$ should satisfy

$$Ls = \lambda \left( s - \frac{n_1 - n_2}{n} 1 \right)$$
Graph Laplacian

- Define $\mathbf{x} = \mathbf{s} - \frac{n_1 - n_2}{n} \mathbf{1}$. Then

$$L\mathbf{x} = L\mathbf{s} - \frac{n_1 - n_2}{n} L\mathbf{1}$$

$$= L\mathbf{s} \quad \text{since } L\mathbf{1} = \mathbf{0}$$

$$= \lambda \left( \mathbf{s} - \frac{n_1 - n_2}{n} \mathbf{1} \right)$$

$$= \lambda \mathbf{x}$$

And

$$\mathbf{x}^T L\mathbf{x} = \mathbf{s}^T L\mathbf{s} - 2 \frac{n_1 - n_2}{n} \mathbf{s}^T L\mathbf{1} + \frac{n_1 - n_2}{n})^2 \mathbf{1}^T L\mathbf{1} = \mathbf{s}^T L\mathbf{s}.$$
Graph Laplacian

Now

\[ x^T L x = \lambda x^T x \]

And

\[ x^T x = \left( s - \frac{n_1 - n_2}{n} \mathbf{1} \right)^T \left( s - \frac{n_1 - n_2}{n} \mathbf{1} \right) \]

\[ = s^T s + \frac{(n_1 - n_2)^2}{n^2} \mathbf{1}^T \mathbf{1} - 2 \frac{n_1 - n_2}{n} s^T \mathbf{1} \]

\[ = n + \frac{(n_1 - n_2)^2}{n} - 2(n_1 - n_2) \]

\[ = \frac{1}{n} \left( n^2 + (n_1 - n_2)^2 - 2n(n_1 - n_2) \right) \]

\[ = \frac{1}{n} (n - n_1 + n_2)^2 = \frac{4n_2^2}{n}. \]
Graph Laplacian

In summary, find eigenvector $\mathbf{x}$ of $L$ such that

So that $\|\mathbf{x}\|_2^2 = 4n^2/n$, and
$L\mathbf{x} = \lambda \mathbf{x}$ with minimal possible $\lambda$

Observe that $\lambda \geq 0$ since $\mathbf{x}^T L \mathbf{x} \geq 0$ for all $\mathbf{x} \in \mathbb{R}^n$

Now, $\mathbf{1}$ is an eigenvector of $L$ with eigenvalue 0

$$
\mathbf{1}^T L \mathbf{1} = \sum_{ij} L_{ij} = \sum_{ij} D_{ij} - \sum_{ij} A_{ij}
$$

$$
= \sum_i k_i - \sum_{ij} A_{ij} = 2m - 2m = 0
$$

where $m$ is the number of edges
Thus, the smallest eigenvalue is 0 with eigenvector $1$

It gives no partition

Therefore, we need to look for the second smallest eigenvector

Let $v \in \mathbb{R}^n$ be the eigenvector of $L$

corresponding to the second smallest eigenvalue

Final step: find $s \in \{-1, 1\}^n$

That is most aligned with $v$

So that $|\{i : s_i = +1\}| = n_1$, $|\{i : s_i = -1\}| = n_2$
Spectral clustering

Alignment of $s$ and $v$: $\sum_i s_i v_i$

To maximize alignment
Order $[n]$ as per the decreasing order of values of $v$
For top $n_1$ indices, assign corresponding $s_i = +1$ and rest $-1$

Minor subtlety:
Both $v$ and $-v$ are eigenvectors
Therefore, repeat the above with respect to $-v$
Choose the partition that minimizes our objective
Modularity maximization

Spectral clustering (or partition)
- Uses informative eigenvector of Graph Laplacian
- Requires number of nodes in the partition

Modularity maximization
- Provides a natural way to determine partitions, their sizes
- A variation on spectral partitioning

So what is Modularity?
- Let $c_i$ denote cluster index of node $i$; $\mathbf{c} = (c_i)$
- Then modularity of clustering $\mathbf{c}$, denoted as $Q(\mathbf{c})$ is

$$Q(\mathbf{c}) = \frac{1}{2m} \sum_{\substack{i \in N \ \ j \in N : c_i = c_j}} \left( A_{ij} - \frac{k_i k_j}{2m} \right)$$
Modularity

Modularity of clustering $c$

$$Q(c) = \frac{1}{2m} \sum_{i \in N, j \in N: c_i = c_j} \left( A_{ij} - \frac{k_i k_j}{2m} \right)$$

$k_i k_j / 2m$ captures the probability that edge between $i, j$ exists under “random graph formation”

that is, modularity tries to capture deviation from it

Modularity maximization

find clustering $c$ so that $Q(c)$ is maximized
find maximal deviation from “random graph”
we will restrict to $c \in \{-1, 1\}^n$
Modularity maximization

Modularity for $c \in \{-1, 1\}^n$

$$Q(c) = \frac{1}{2m} \sum_{i \in N \atop j \in N: c_i = c_j} \left( A_{ij} - \frac{k_i k_j}{2m} \right)$$

$$= \frac{1}{2m} \sum_{ij} \left( A_{ij} - \frac{k_i k_j}{2m} \right) \left( 1 + c_i c_j \right) / 2$$

$$= \frac{1}{4m} \sum_{ij} B_{ij} \left( 1 + c_i c_j \right)$$

where $B_{ij} = A_{ij} - \frac{k_i k_j}{2m}$, or $B = A - \frac{1}{2m} kk^T$ with $k = (k_i)$

Therefore

$$4mQ(c) = \sum_{ij} B_{ij} + \sum_{ij} c_i B_{ij} c_j = 1^T B 1 + c^T B c.$$
Modularity maximization

Now

\[ 1^T B 1 = 1^T A 1 - \frac{1}{2m} 1^T k k^T 1 \]

\[ = 2m - \frac{1}{2m} 2m \times 2m = 0, \]

Therefore

\[ 4m Q(c) = c^T B c \]

Therefore, modularity maximization can be relaxed

\[ \text{maximize } c^T B c \text{ such that } c^T c = n \]
Modularity maximization

Consider Lagrangian formulation

\[ g(\lambda, \mathbf{c}) = \mathbf{c}^T \mathbf{B} \mathbf{c} + \lambda(n - \mathbf{c}^T \mathbf{c}). \]

Taking partial derivative of \( g \) with respect to \( c_i \)

\[ \frac{\partial g}{\partial c_i} = 2 \sum_j B_{ij} c_j - 2 \lambda c_i \]

Setting above to 0 for all \( i \) yields

\[ \mathbf{B} \mathbf{c} = \lambda \mathbf{c} \]

The corresponding modularity is

\[ Q(\mathbf{c}) = \frac{1}{4m} \lambda \mathbf{c}^T \mathbf{c} = \lambda \frac{n}{2m}. \]
Modularity maximization

In summary

Find largest eigenvector, \( \mathbf{v} \), of \( B \) to maximize modularity
Use \( \mathbf{v} \) to find partition \( \mathbf{c} \)

Final step

Find \( \mathbf{c} \in \{-1, 1\}^n \) that maximizes alignment with \( \mathbf{v} \)
That is, \( \sum_i c_i v_i \) is maximum
This has a simple answer

\[
c_i = \begin{cases} 
1 & \text{if } v_i \geq 0 \\
-1 & \text{if } v_i < 0.
\end{cases}
\]
Generalizations

We have discussed clustering with two clusters

It can be applied repeatedly to obtain multiple / hierarchical clusters
Or, Modularity optimization can be defined for more than two clusters
But solving it can be computationally hard

A generic *Spectral approach*

Let $W = [W_{ij}]$ be matrix (usually symmetric) of interest
Find top $k$ eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_k$ of $W$

"top" = smallest or largest eigenvalues/vectors depending on context

Assign $(\mathbf{v}_1)_i, \ldots, (\mathbf{v}_k)_i \in \mathbb{R}^k$ to node $i \in \mathcal{N}$

embedding of each node $i$ in $k$ dimensional space

Use this embedding to do further *processing*