§2 Clustering

Qin Zhang
Motivations

- Group together similar documents/webpages/images/people/proteins/products
- One of the most important problems in machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics.
- Cluster topic/articles from the web by “same story”.
- Google image
- Many others ...
Motivations

- Group together similar documents/webpages/images/people/proteins/products
- One of the most important problems in machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics.
- Cluster topic/articles from the web by “same story”.
- Google image
- Many others ...

- Given a cloud of data points we want to understand their structures
- Community discovery in social networks
Motivations

- Group together similar
documents/webpages/images/people/proteins/products

- One of the most important problems in machine learning,
  pattern recognition, image analysis, information retrieval,
  and bioinformatics.

- Community discovery in social networks
Clustering is an extremely broad and ill-defined topic. There are many many variants; we will not try to cover all of them. Instead we will focus on three broad classes of algorithms

- Hierarchical Clustering
- Assignment-based Clustering
- Spectral Clustering
Clustering is an extremely broad and ill-defined topic. There are many many variants; we will not try to cover all of them. Instead we will focus on three broad classes of algorithms

- Hierarchical Clustering
- Assignment-based Clustering
- Spectral Clustering

There is a saying

“When data is easily cluster-able, most clustering algorithms work quickly and well. When is not easily cluster-able, then no algorithm will find good clusters.”
Hierarchical Clustering
**Problem:** Start with a set $X \subseteq \mathbb{R}^d$, and a metric $d : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_+$. A *cluster* $C$ is a subset of $X$. A *clustering* is a partition $\rho(X) = \{C_1, C_2, \ldots, C_k\}$, where

1. Each $C_i \subset X$.
2. Each pair $C_i \cap C_j = \emptyset$.
3. Each $\bigcup_{i=1}^k C_i = X$. 
Problem: Start with a set $X \subset \mathbb{R}^d$, and a metric $d : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$. A cluster $C$ is a subset of $X$. A clustering is a partition $\rho(X) = \{C_1, C_2, \ldots, C_k\}$, where

1. Each $C_i \subset X$.
2. Each pair $C_i \cap C_j = \emptyset$.
3. Each $\bigcup_{i=1}^k C_i = X$.

Goal:

1. (close intra) For each $C \in \rho(X)$ for all $x, x' \in C$, $d(x, x')$ is small.
2. (far inter) For each $C_i, C_j \in \rho(X)$ ($i \neq j$), for most $x_i \in C_i$ and $x_j \in C_j$, $d(x_i, x_j)$ is large.
Hierarchical Clustering

1. Each $x_i \in X$ forms a cluster $C_i$
2. While exists two clusters close enough
   (a) Find the closest two clusters $C_i, C_j$.
   (b) Merge $C_i, C_j$ into a single cluster
Hierarchical Clustering

1. Each $x_i \in X$ forms a cluster $C_i$
2. While exists two clusters close enough
   (a) Find the closest two clusters $C_i, C_j$.
   (b) Merge $C_i, C_j$ into a single cluster

Two things to define

1. Define a *distance between clusters*.
2. Define *close enough*
Definition of “distance between clusters”

- Distance between center of clusters. What does center mean?
Definition of “distance between clusters”

- Distance between center of clusters. What does center mean?
  - the mean (average) point. Called the centroid.
Definition of “distance between clusters”

- Distance between center of clusters. What does center mean?
  - the mean (average) point. Called the centroid.
  - the center-point (like a high-dimensional median)
Definition of “distance between clusters”

- Distance between center of clusters. What does center mean?
  - the mean (average) point. Called the centroid.
  - the center-point (like a high-dimensional median)
  - center of the MEB (minimum enclosing ball)
Definition of “distance between clusters”

- Distance between center of clusters. What does center mean?
  - the mean (average) point. Called the centroid.
  - the center-point (like a high-dimensional median)
  - center of the MEB (minimum enclosing ball)
  - some (random) representative point
Definition of “distance between clusters”

- Distance between center of clusters. What does center mean?
  - the mean (average) point. Called the centroid.
  - the center-point (like a high-dimensional median)
  - center of the MEB (minimum enclosing ball)
  - some (random) representative point

- Distance between closest pair of points
Definition of “distance between clusters”

- Distance between center of clusters. What does center mean?
  - the mean (average) point. Called the centroid.
  - the center-point (like a high-dimensional median)
  - center of the MEB (minimum enclosing ball)
  - some (random) representative point

- Distance between closest pair of points

- Distance between furthest pair of points
Definition of “distance between clusters”

- Distance between center of clusters. What does center mean?
  - the mean (average) point. Called the centroid.
  - the center-point (like a high-dimensional median)
  - center of the MEB (minimum enclosing ball)
  - some (random) representative point

- Distance between closest pair of points

- Distance between furthest pair of points

- Average distance between all pairs of points in different clusters
Definition of “distance between clusters”

- Distance between center of clusters. What does center mean?
  - the mean (average) point. Called the centroid.
  - the center-point (like a high-dimensional median)
  - center of the MEB (minimum enclosing ball)
  - some (random) representative point

- Distance between closest pair of points

- Distance between furthest pair of points

- Average distance between all pairs of points in different clusters

- Radius of minimum enclosing ball of joined cluster
Definition of “close enough”

- If information is known about the data, perhaps some absolute value \( \tau \) can be given.
Definition of “close enough”

- If information is known about the data, perhaps some absolute value $\tau$ can be given.
- If the {diameter, or radius of MEB, or average distance from center point} is beneath some threshold. This fixes the scale of a cluster, which could be good or bad.
Definition of “close enough”

- If information is known about the data, perhaps some absolute value $\tau$ can be given.
- If the $\{\text{diameter, or radius of MEB, or average distance from center point}\}$ is beneath some threshold. This fixes the scale of a cluster, which could be good or bad.
- If the density is beneath some threshold; where density is $\# \text{ points/volume or } \# \text{ points/radius}^d$
Definition of “close enough”

- If information is known about the data, perhaps some absolute value $\tau$ can be given.
- If the \{ diameter, or radius of MEB, or average distance from center point \} is beneath some threshold. This fixes the scale of a cluster, which could be good or bad.
- If the density is beneath some threshold; where density is $\#$ points/volume or $\#$ points/radius$^d$
- If the joint density decrease too much from single cluster density. Variations of this are called the “elbow” technique.
Definition of “close enough”

- If information is known about the data, perhaps some absolute value $\tau$ can be given.

- If the \{diameter, or radius of MEB, or average distance from center point\} is beneath some threshold. This fixes the scale of a cluster, which could be good or bad.

- If the density is beneath some threshold; where density is $\#$ points/volume or $\#$ points/radius$^d$.

- If the joint density decrease too much from single cluster density. Variations of this are called the “elbow” technique.

- When the number of clusters is $k$ (for some magical value $k$).
Definition of “close enough”

- If information is known about the data, perhaps some absolute value $\tau$ can be given.
- If the $\{\text{diameter, or radius of MEB, or average distance from center point}\}$ is beneath some threshold. This fixes the scale of a cluster, which could be good or bad.
- If the density is beneath some threshold; where density is $\#\text{ points/volume or } \#\text{ points/radius}^d$
- If the joint density decrease too much from single cluster density. Variations of this are called the “elbow” technique.
- When the number of clusters is $k$ (for some magical value $k$).

**Example** (on board): Distance is distance between centroids. Stop when there is 1 cluster left.
Assignment-based Clustering

\((k\text{-center, } k\text{-mean, } k\text{-median})\)
Definitions

For a set $X$, and distance $d : X \times X \to \mathbb{R}_+$, the output is a set of points $C = \{c_1, \ldots, c_k\}$, which implicitly defines a set of clusters where $\phi_C(x) = \arg \min_{c \in C} d(x, c)$. 
Assignment based clustering

Definitions

For a set $X$, and distance $d : X \times X \rightarrow \mathbb{R}_+$, the output is a set of points $C = \{c_1, \ldots, c_k\}$, which implicitly defines a set of clusters where $\phi_C(x) = \arg\min_{c \in C} d(x, c)$.

- The *$k$-center cluster problem* is to find the set $C$ of $k$ clusters (often, but not always as a subset of $X$) to minimize $\max_{x \in X} d(\phi_C(x), x)$. 

Assignment based clustering

Definitions

For a set $X$, and distance $d : X \times X \to \mathbb{R}_+$, the output is a set of points $C = \{c_1, \ldots, c_k\}$, which implicitly defines a set of clusters where $\phi_C(x) = \arg \min_{c \in C} d(x, c)$.

- The *k-center cluster problem* is to find the set $C$ of $k$ clusters (often, but not always as a subset of $X$) to

  $$\text{minimize} \quad \max_{x \in X} d(\phi_C(x), x).$$

- The *k-means cluster problem*: minimize

  $$\sum_{x \in X} d(\phi_C(x), x)^2$$
Assignment based clustering

Definitions

For a set $X$, and distance $d : X \times X \rightarrow \mathbb{R}_+$, the output is a set of points $C = \{c_1, \ldots, c_k\}$, which implicitly defines a set of clusters where $\phi_C(x) = \arg \min_{c \in C} d(x, c)$.

- The $k$-center cluster problem is to find the set $C$ of $k$ clusters (often, but not always as a subset of $X$) to minimize $\max_{x \in X} d(\phi_C(x), x)$.

- The $k$-means cluster problem: minimize $\sum_{x \in X} d(\phi_C(x), x)^2$

- The $k$-median cluster problem: minimize $\sum_{x \in X} d(\phi_C(x), x)$
**Gonzalez Algorithm** A simple greedy algorithm for $k$-center

1. Choose $c_1 \in X$ arbitrarily. Let $S_1 = \{c_1\}$
2. For $i = 2$ to $k$ do
   
   (a) Set $c_i = \arg\max_{x \in X} d(x, \phi_{S_{i-1}}(x))$
   (b) Let $S_i = \{c_1, \ldots, c_i\}$
Gonzalez Algorithm A simple greedy algorithm for $k$-center

1. Choose $c_1 \in X$ arbitrarily. Let $S_1 = \{c_1\}$
2. For $i = 2$ to $k$ do
   (a) Set $c_i = \arg \max_{x \in X} d(x, \phi_{S_{i-1}}(x))$
   (b) Let $S_i = \{c_1, \ldots, c_i\}$

In the worst case, it gives a 2-approximation to the optimal solution.
**Gonzalez Algorithm** A simple greedy algorithm for \( k \)-center

1. Choose \( c_1 \in X \) arbitrarily. Let \( S_1 = \{c_1\} \)
2. For \( i = 2 \) to \( k \) do
   
   (a) Set \( c_i = \arg \max_{x \in X} d(x, \phi_{S_{i-1}}(x)) \)
   (b) Let \( S_i = \{c_1, \ldots, c_i\} \)

In the worst case, it gives a 2-approximation to the optimal solution.

**Analysis** (on board)
Parallel Guessing Algorithm for $k$-center

Run for $R = (1 + \epsilon/2), (1 + \epsilon/2)^2, \ldots, \Delta$

1. We pick an arbitrary point as the first center, $C = \{c_1\}$.
2. For each point $p \in P$ compute $r_p = \min_{c \in C} d(p, c)$ and test whether $r_p > R$. If it is, then set $C = C \cup \{p\}$.
3. Abort at $|C| > k$

Pick the minimum $R$ that does not abort.
Parallel Guessing Algorithm for $k$-center

Run for $R = (1 + \epsilon/2), (1 + \epsilon/2)^2, \ldots, \Delta$

1. We pick an arbitrary point as the first center, $C = \{c_1\}$.
2. For each point $p \in P$ compute $r_p = \min_{c \in C} d(p, c)$ and test whether $r_p > R$. If it is, then set $C = C \cup \{p\}$.
3. Abort at $|C| > k$

Pick the minimum $R$ that does not abort.

In the worst case, it gives a $(2 + \epsilon)$-approximation to the optimal solution (on board).
Parallel Guessing Algorithm for $k$-center

Run for $R = (1 + \epsilon/2), (1 + \epsilon/2)^2, \ldots, \Delta$

1. We pick an arbitrary point as the first center, $C = \{c_1\}$.
2. For each point $p \in P$ compute $r_p = \min_{c \in C} d(p, c)$ and test whether $r_p > R$. If it is, then set $C = C \cup \{p\}$.
3. Abort at $|C| > k$

Pick the minimum $R$ that does not abort.

In the worst case, it gives a $(2 + \epsilon)$-approximation to the optimal solution (on board).

Can be implemented as a streaming algorithm – performs a linear scan of the data, uses space $\tilde{O}(k)$ and time $\tilde{O}(nk)$. 

Parallel Guessing Algorithm
Lloyd’s Algorithm

Lloyd’s Algorithm The well-known algorithm for $k$-means

1. Choose $k$ points $C \subset X$ (arbitrarily?)
2. Repeat
   (a) For all $x \in X$, find $\phi_C(x)$ (closest center $c \in C$ to $x$)
   (b) For all $i \in [k]$, let $c_i = \text{average}\{ x \in X \mid \phi_C(x) = c_i \}$
3. until the set $C$ is unchanged

http://www.math.le.ac.uk/people/ag153/homepage/KmeansKmedoids/Kmeans_Kmedoids.html
Lloyd’s Algorithm

The well-known algorithm for $k$-means

1. Choose $k$ points $C \subset X$ (arbitrarily?)
2. Repeat
   (a) For all $x \in X$, find $\phi_C(x)$ (closest center $c \in C$ to $x$)
   (b) For all $i \in [k]$, let $c_i = \text{average}\{x \in X \mid \phi_C(x) = c_i\}$
3. until the set $C$ is unchanged

If the main loop has $R$ rounds, then the running time is $O(Rnk)$
Lloyd’s Algorithm The well-known algorithm for $k$-means

1. Choose $k$ points $C \subset X$ (arbitrarily?)
2. Repeat
   (a) For all $x \in X$, find $\phi_C(x)$ (closest center $c \in C$ to $x$)
   (b) For all $i \in [k]$, let $c_i = \text{average}\{x \in X \mid \phi_C(x) = c_i\}$
3. until the set $C$ is unchanged

http://www.math.le.ac.uk/people/ag153/homepage/KmeansKmedoids/Kmeans_Kmedoids.html

If the main loop has $R$ rounds, then the running time is $O(Rnk)$

But
1. What is $R$? Converge in finite number of steps?
2. How do we initialize $C$?
3. How accurate is this algorithm?
It is finite. The cost $\sum_{x \in X} d(x, \phi_C(x))^2$ is always decreasing, and there are a finite number of possible distinct cluster centers. But it could be exponential in $k$ and $d$ (the dimension when Euclidean distance used): $n^{O(kd)}$ (number of voronoi cell)
Value of $R$

- It is finite. The cost $\sum_{x \in X} d(x, \phi_C(x))^2$ is always decreasing, and there are a finite number of possible distinct cluster centers. But it could be exponential in $k$ and $d$ (the dimension when Euclidean distance used): $n^{O(kd)}$ (number of voronoi cell).

- However, usually $R = 10$ is fine.
Value of $R$

- It is finite. The cost $\sum_{x \in X} d(x, \phi_C(x))^2$ is always decreasing, and there are a finite number of possible distinct cluster centers. But it could be exponential in $k$ and $d$ (the dimension when Euclidean distance used): $n^{O(kd)}$ (number of voronoi cell).

- However, usually $R = 10$ is fine.

- Smoothed analysis: if data perturbed randomly slightly, then $R = O(n^{35}k^{34}d^{8})$. This is “polynomial”, but still ridiculous.
It is finite. The cost $\sum_{x \in X} d(x, \phi_C(x))^2$ is always decreasing, and there are a finite number of possible distinct cluster centers. But it could be exponential in $k$ and $d$ (the dimension when Euclidean distance used): $n^{O(kd)}$ (number of voronoi cell)

However, usually $R = 10$ is fine.

Smoothed analysis: if data perturbed randomly slightly, then $R = O(n^{35} k^{34} d^8)$. This is “polynomial”, but still ridiculous.

If all points are on a grid of length $M$, then $R = O(dn^4 M^2)$. But thats still way too big.
Value of $R$

- It is finite. The cost $\sum_{x \in X} d(x, \phi_C(x))^2$ is always decreasing, and there are a finite number of possible distinct cluster centers. But it could be exponential in $k$ and $d$ (the dimension when Euclidean distance used): $n^{O(kd)}$ (number of voronoi cell).

- However, usually $R = 10$ is fine.

- Smoothed analysis: if data perturbed randomly slightly, then $R = O(n^{35} k^{34} d^8)$. This is “polynomial”, but still ridiculous.

- If all points are on a grid of length $M$, then $R = O(dn^4 M^2)$. But thats still way too big.

**Conclusion:** There are crazy special cases that can take a long time, but usually it works.
Initialize $C$

- Random set of points. By coupon collectors, we know that we need about $O(k \log k)$ to get one in each cluster, given that each cluster contains equal number of points. We can later reduce to $k$ clusters, by merging extra clusters.
Random set of points. By coupon collectors, we know that we need about $O(k \log k)$ to get one in each cluster, given that each cluster contains equal number of points. We can later reduce to $k$ clusters, by merging extra clusters.

Randomly partition $X = \{X_1, X_2, \ldots, X_k\}$ and take $c_i = \text{average}(X_i)$. This biases towards “center” of $X$. 
Initialize $C$

- Random set of points. By coupon collectors, we know that we need about $O(k \log k)$ to get one in each cluster, given that each cluster contains equal number of points. We can later reduce to $k$ clusters, by merging extra clusters.

- Randomly partition $X = \{X_1, X_2, \ldots, X_k\}$ and take $c_i = \text{average}(X_i)$. This biases towards “center” of $X$.

- Gonzalez algorithm (for $k$-center). This biases too much to outlier points.
- Can be arbitrarily bad. (Give an example?)
Can be arbitrarily bad. (Give an example?)

4 vertices of a rectangle (width $\gg$ height)
- Can be arbitrarily bad. (Give an example?)

- Theory algorithm: Gets \((1 + \epsilon)\)-approximation for \(k\)-means in \(2^{(k/\epsilon)^{O(1)}} nd\) time. (Kumar, Sabharwal, Sen 2004)
Accuracy

- Can be arbitrarily bad. (Give an example?)

- Theory algorithm: Gets \((1 + \epsilon)\)-approximation for \(k\)-means in \(2^{(k/\epsilon)^{O(1)}} nd\) time. (Kumar, Sabharwal, Sen 2004)

- The following \(k\)-means++ is \(O(\log k)\)-approximation \(^a\).

\(^a\)Ref: \(k\)-means++: The Advantages of Careful Seeding

\(k\)-means++

1. Choose \(c_1 \in X\) arbitrarily. Let \(S_1 = \{c_1\}\)
2. For \(i = 2\) to \(k\) do
   
   (a) Choose \(c_i\) from \(X\) with probability proportional to \(d(x, \phi_{S_{i-1}}(x))^2\)
   (b) Let \(S_i = \{c_1, \ldots, c_i\}\)
The key step that makes Lloyd's algorithm so cool is

$$\text{average}\{x \in X\} = \arg\min_{c \in \mathbb{R}^d} \sum_{x \in X} \|c - x\|_2^2.$$  

But it only works for distance function $$d(x, c) = \|x - c\|_2$$

Is effected by outliers more than $$k$$-median clustering.
Local Search Algorithm  Designed for $k$-median

1. Choose $K$ centers at random from $X$. Call this set $C$.
2. Repeat
   For each $u$ in $X$ and $c$ in $C$, compute
   $\text{Cost}(C - \{c\} + \{u\})$. Find $(u, c)$ for which this cost is smallest, and replace $C$ with $C - \{c\} + \{u\}$.
3. Stop when Cost($C$) does not change significantly
Local Search Algorithm Designed for $k$-median

1. Choose $K$ centers at random from $X$. Call this set $C$.
2. Repeat
   For each $u$ in $X$ and $c$ in $C$, compute
   \[
   \text{Cost}(C - \{c\} + \{u\}).
   \]
   Find $(u, c)$ for which this cost is smallest, and replace
   $C$ with $C - \{c\} + \{u\}$.
3. Stop when Cost($C$) does not change significantly

Can be used to design a 5-approximation algorithm \(^a\).

\(^a\)Ref: Local Search Heuristics for k-median and Facility Location Problems

Q: How can you speed up the swap operation by avoiding searching over all $(u, c)$ pairs?
Spectral Clustering
Top-down Clustering Framework

1. Find the best cut of the data into two pieces.
2. Recur on both pieces until that data should not be split anymore.
Top-down Clustering Framework

1. Find the best cut of the data into two pieces.
2. Recur on both pieces until that data should not be split anymore.

What is the best way to partition the data?
This is a big question. Let’s first talk about graphs.
**Graph:** $G = (V, E)$ is defined by a set of vertices $V = \{v_1, v_2, \ldots, v_n\}$ and a set of edges $E = \{e_1, e_2, \ldots, e_m\}$ where each edge $e_j$ is an unordered (or ordered in a directed graph) pair of edges $e_j = \{v_i, v_{i'}\}$ (or $e_j = (v_i, v_{i'})$).

Example:
**Graphs**

**Graph:** \( G = (V, E) \) is defined by a set of vertices \( V = \{v_1, v_2, \ldots, v_n\} \) and a set of edges \( E = \{e_1, e_2, \ldots, e_m\} \) where each edge \( e_j \) is an unordered (or ordered in a directed graph) pair of edges \( e_j = \{v_i, v_i'\} \) (or \( e_j = (v_i, v_i') \)).

**Example:**

Adjacent list representation

\[
A = \begin{pmatrix}
0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
\end{pmatrix}
\]
From point sets to graphs

- $\epsilon$-neighborhood graph: Connect all points whose pairwise distance are smaller than $\epsilon$. An unweighted graph
- **$\epsilon$-neighborhood graph**: Connect all points whose pairwise distance are smaller than $\epsilon$. An unweighted graph.

- **$k$-nearest neighbor graphs**: Connect $v_i$ with $v_j$ if $v_j$ is among the $k$-nearest neighbors of $v_i$. Naturally a directed graph. Two ways to make it undirected: (1) neglect the direction; (2) mutual $k$-nearest neighbor.
From point sets to graphs

- **$\epsilon$-neighborhood graph**: Connect all points whose pairwise distance are smaller than $\epsilon$. An unweighted graph.

- **$k$-nearest neighbor graphs**: Connect $v_i$ with $v_j$ if $v_j$ is among the $k$-nearest neighbors of $v_i$. Naturally a directed graph. Two ways to make it undirected: (1) neglect the direction; (2) mutual $k$-nearest neighbor.

- **The fully connected graph**: Connect all points with finite distances with each other, and weight all edges by $d(v_i, v_j)$, where $d(v_i, v_j)$ is the distance between $v_i$ and $v_j$. 
A good cut

Rule of the thumb

1. Many edges in a cluster.
   
   *Volume* if a cluster is $\text{Vol}(S) = \text{the number of edges with at least one vertex in } V$.

2. Few edges between clusters.
   
   *Cut* between two clusters $S, T$ is $\text{Cut}(S, T) = \text{the number of edges with one vertex in } S \text{ and the other in } T$. 
A good cut

Rule of the thumb

1. Many edges in a cluster.

   \textit{Volume} if a cluster is $\text{Vol}(S) =$ the number of edges with at least one vertex in $V$.

2. Few edges between clusters.

   \textit{Cut} between two clusters $S, T$ is $\text{Cut}(S, T) =$ the number of edges with one vertex in $S$ and the other in $T$.

**Ratio cut:** $\text{RatioCut}(S, T) = \frac{\text{Cut}(S, T)}{|S|} + \frac{\text{Cut}(S, T)}{|T|}$

**Normalized cut:** $\text{NCut}(S, T) = \frac{\text{Cut}(S, T)}{\text{Vol}(S)} + \frac{\text{Cut}(S, T)}{\text{Vol}(T)}$
A good cut

Rule of the thumb
1. Many edges in a cluster.

   \( Volume \) if a cluster is \( Vol(S) = \) the number of edges with at least one vertex in \( V \).

2. Few edges between clusters.

   \( Cut \) between two clusters \( S, T \) is \( Cut(S, T) = \) the number of edges with one vertex in \( S \) and the other in \( T \).

Ratio cut: \( \text{RatioCut}(S, T) = \frac{Cut(S, T)}{|S|} + \frac{Cut(S, T)}{|T|} \)

Normalized cut: \( \text{NCut}(S, T) = \frac{Cut(S, T)}{Vol(S)} + \frac{Cut(S, T)}{Vol(T)} \)
1. *Adjacent* matrix of the graph $A$
2. *Degree* (diagonal) matrix $D$
3. *Laplacian* matrix $L = D - A$
4. Eigenvector of a matrix $M$ is the vector $v$ such that $Mv = \lambda v$, where $\lambda$ is a scalar, called the corresponding *eigenvalue*. 
1. *Adjacent* matrix of the graph \( A \)
2. *Degree* (diagonal) matrix \( D \)
3. *Laplacian* matrix \( L = D - A \)
4. Eigenvector of a matrix \( M \) is the vector \( v \) such that \( Mv = \lambda v \), where \( \lambda \) is a scalar, called the corresponding *eigenvalue*.

Adjacent list matrix

\[
A = \begin{pmatrix}
0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
\end{pmatrix}
\]

Degree matrix

\[
D = \begin{pmatrix}
3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{pmatrix}
\]
Laplacian matrix and eigenvector (cont.)

\[
L = D - A = \begin{pmatrix}
3 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\
-1 & 2 & 0 & -1 & 0 & 0 & 0 & 0 \\
-1 & 0 & 3 & -1 & -1 & 0 & 0 & 0 \\
-1 & -1 & -1 & 3 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 3 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & -1 & 3 & -1 & -1 \\
0 & 0 & 0 & 0 & -1 & -1 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \\
\end{pmatrix}
\]
Laplacian matrix and eigenvector (cont.)

\[ L = D - A = \begin{pmatrix}
3 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\
-1 & 2 & 0 & -1 & 0 & 0 & 0 & 0 \\
-1 & 0 & 3 & -1 & -1 & 0 & 0 & 0 \\
-1 & -1 & -1 & 3 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 3 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & -1 & 3 & -1 & -1 \\
0 & 0 & 0 & 0 & -1 & -1 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \\
\end{pmatrix} \]

Eigenvalues and eigenvectors (after normalization) of Laplacian matrix

<table>
<thead>
<tr>
<th>( \lambda )</th>
<th>0</th>
<th>0.278</th>
<th>1.11</th>
<th>2.31</th>
<th>3.46</th>
<th>4</th>
<th>4.82</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/( \sqrt{8} )</td>
<td>-.36</td>
<td>0.08</td>
<td>0.10</td>
<td>0.28</td>
<td>0.25</td>
<td>1/( \sqrt{2} )</td>
<td></td>
</tr>
<tr>
<td>1/( \sqrt{8} )</td>
<td>-.42</td>
<td>0.18</td>
<td>0.64</td>
<td>-.38</td>
<td>0.25</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1/( \sqrt{8} )</td>
<td>-.20</td>
<td>-.11</td>
<td>0.61</td>
<td>0.03</td>
<td>-.25</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1/( \sqrt{8} )</td>
<td>-.36</td>
<td>0.08</td>
<td>0.10</td>
<td>0.28</td>
<td>0.25</td>
<td>-1/( \sqrt{2} )</td>
<td></td>
</tr>
<tr>
<td>1/( \sqrt{8} )</td>
<td>0.17</td>
<td>-.37</td>
<td>0.21</td>
<td>-.54</td>
<td>-.25</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1/( \sqrt{8} )</td>
<td>0.36</td>
<td>-.08</td>
<td>-.10</td>
<td>-.28</td>
<td>0.75</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1/( \sqrt{8} )</td>
<td>0.31</td>
<td>-.51</td>
<td>-.36</td>
<td>-.56</td>
<td>0.56</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1/( \sqrt{8} )</td>
<td>0.50</td>
<td>0.73</td>
<td>0.08</td>
<td>0.11</td>
<td>0.11</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
Laplacian matrix and eigenvector (cont.)

\[
L = D - A = \begin{pmatrix}
3 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\
-1 & 2 & 0 & -1 & 0 & 0 & 0 & 0 \\
-1 & 0 & 3 & -1 & -1 & 0 & 0 & 0 \\
-1 & -1 & -1 & 3 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 3 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & -1 & 3 & -1 & -1 \\
0 & 0 & 0 & 0 & -1 & -1 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 1
\end{pmatrix}
\]

Eigenvalues and eigenvectors (after normalization) of Laplacian matrix

<table>
<thead>
<tr>
<th>(\lambda)</th>
<th>0</th>
<th>1.11</th>
<th>2.31</th>
<th>3.46</th>
<th>4</th>
<th>4.82</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1/\sqrt{8})</td>
<td>(-.36)</td>
<td>0.08</td>
<td>0.10</td>
<td>0.28</td>
<td>0.25</td>
<td>1/\sqrt{2}</td>
</tr>
<tr>
<td>(1/\sqrt{8})</td>
<td>(-.42)</td>
<td>0.18</td>
<td>0.64</td>
<td>(-.38)</td>
<td>0.25</td>
<td>0</td>
</tr>
<tr>
<td>(1/\sqrt{8})</td>
<td>(-.20)</td>
<td>(-.11)</td>
<td>0.61</td>
<td>0.03</td>
<td>(-.25)</td>
<td>0</td>
</tr>
<tr>
<td>(1/\sqrt{8})</td>
<td>(-.36)</td>
<td>0.08</td>
<td>0.10</td>
<td>0.28</td>
<td>0.25</td>
<td>(-1/\sqrt{2})</td>
</tr>
<tr>
<td>(1/\sqrt{8})</td>
<td>0.17</td>
<td>(-.37)</td>
<td>0.21</td>
<td>(-.54)</td>
<td>(-.25)</td>
<td>0</td>
</tr>
<tr>
<td>(1/\sqrt{8})</td>
<td>0.36</td>
<td>(-.08)</td>
<td>(-.10)</td>
<td>(-.28)</td>
<td>0.75</td>
<td>0</td>
</tr>
<tr>
<td>(1/\sqrt{8})</td>
<td>0.31</td>
<td>(-.51)</td>
<td>(-.36)</td>
<td>(-.56)</td>
<td>0.56</td>
<td>0</td>
</tr>
<tr>
<td>(1/\sqrt{8})</td>
<td>0.50</td>
<td>0.73</td>
<td>0.08</td>
<td>0.11</td>
<td>0.11</td>
<td>0</td>
</tr>
</tbody>
</table>

(Graph embedding on board)
Laplacian matrix and eigenvector (cont.)

\[ \mathbf{L} = \mathbf{D} - \mathbf{A} = \begin{pmatrix} 3 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 2 & 0 & -1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 3 & -1 & -1 & 0 & 0 & 0 \\ -1 & -1 & -1 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 3 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 3 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 & -1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix} \]

Eigenvalues and eigenvectors (after normalization) of Laplacian matrix

<table>
<thead>
<tr>
<th>( \lambda )</th>
<th>0</th>
<th>0.278</th>
<th>1.11</th>
<th>2.31</th>
<th>3.46</th>
<th>4</th>
<th>4.82</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 1/\sqrt{8} )</td>
<td>-0.36</td>
<td>0.08</td>
<td>0.10</td>
<td>0.28</td>
<td>0.25</td>
<td>1/\sqrt{2}</td>
<td></td>
</tr>
<tr>
<td>( 1/\sqrt{8} )</td>
<td>-0.42</td>
<td>0.18</td>
<td>0.64</td>
<td>-0.38</td>
<td>0.25</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>( 1/\sqrt{8} )</td>
<td>-0.20</td>
<td>-0.11</td>
<td>0.61</td>
<td>0.03</td>
<td>-0.25</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>( 1/\sqrt{8} )</td>
<td>-0.36</td>
<td>0.08</td>
<td>0.10</td>
<td>0.28</td>
<td>0.25</td>
<td>-1/\sqrt{2}</td>
<td></td>
</tr>
<tr>
<td>( 1/\sqrt{8} )</td>
<td>0.17</td>
<td>-0.37</td>
<td>0.21</td>
<td>-0.54</td>
<td>-0.25</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>( 1/\sqrt{8} )</td>
<td>0.36</td>
<td>-0.08</td>
<td>-0.10</td>
<td>-0.28</td>
<td>0.75</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>( 1/\sqrt{8} )</td>
<td>0.31</td>
<td>-0.51</td>
<td>-0.36</td>
<td>-0.56</td>
<td>0.56</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>( 1/\sqrt{8} )</td>
<td>0.50</td>
<td>0.73</td>
<td>0.08</td>
<td>0.11</td>
<td>0.11</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

(Graph embedding on board)
Spectral Clustering

1. Compute the Laplacian $L$
2. Compute the first $k$ eigenvectors $u_1, \ldots, u_k$ of $L$
3. Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns
4. For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the $i$-th row of $U$.
5. Cluster the points $\{y_1, \ldots, y_n\}$ in $\mathbb{R}^k$ with $k$-means algorithm into clusters $C_1, \ldots, C_k$
The connections

When \( k = 2 \), we want to compute \( \min_{S \subseteq V} \text{RatioCut}(S, T) \)
The connections

When \( k = 2 \), we want to compute \( \min_{S \subset V} \text{RatioCut}(S, T) \)

We can prove that this is equivalent to

\[
\min_{S \subset V} f^T L f \quad \text{s.t.} \quad f \perp 1, \quad \|f\| = 1, \quad \text{and } f_i \text{ is defined as }
\]

\[
f_i = \begin{cases} 
\frac{1}{\sqrt{n}} \cdot \sqrt{\frac{|T|}{|S|}} & \text{if } v_i \in S \\
\frac{1}{\sqrt{n}} \cdot \sqrt{\frac{|S|}{|T|}} & \text{if } v_i \in T = V \setminus S 
\end{cases}
\]
The connections

When $k = 2$, we want to compute $\min_{S \subset V} \text{RatioCut}(S, T)$

We can prove that this is equivalent to

$$\min_{S \subset V} f^T L f \text{ s.t. } f \perp 1, \|f\| = 1, \text{ and } f_i \text{ is defined as}$$

$$f_i = \begin{cases} \frac{1}{\sqrt{n}} \cdot \sqrt{|T|/|S|} & \text{if } v_i \in S \\ \frac{1}{\sqrt{n}} \cdot \sqrt{|S|/|T|} & \text{if } v_i \in T = V \setminus S \end{cases}$$

This is NP-hard. And we solve the following relaxed optimization problem instead.

$$\min_{f \in \mathbb{R}^n} f^T L f \text{ s.t. } f \perp 1, \|f\| = 1$$

The best $f$ is the second smallest eigenvalue of $L$. 
Clustering Social Network
Social network and communities

- Telephone networks
  Communities: groups of people that communicate frequently, e.g., groups of friends, members of a club, etc.
Social network and communities

- Telephone networks
  Communities: groups of people that communicate frequently, e.g., groups of friends, members of a club, etc.

- Email networks
Social network and communities

- Telephone networks
  Communities: groups of people that communicate frequently, e.g., groups of friends, members of a club, etc.

- Email networks

- Collaboration networks: E.g., two people publish paper together are connected.
  Communities: authors working on a particular topic (Wikipedia articles), people working on the same project in Google.
Social network and communities

- Telephone networks
  Communities: groups of people that communicate frequently, e.g., groups of friends, members of a club, etc.

- Email networks

- Collaboration networks: E.g., two people publish paper together are connected.
  Communities: authors working on a particular topic (Wikipedia articles), people working on the same project in Google.

**Goal:** To find the communities. Similar to normal clustering but we sometimes allow overlaps (will not discuss here).
**Difficulty:** Hard to define the “distance”. E.g., if we assign 1 to \((x, y)\) if \(x, y\) are friends and 0 otherwise, then the triangle inequality is not satisfied.
Difficulty: Hard to define the “distance”. E.g., if we assign 1 to \((x, y)\) if \(x, y\) are friends and 0 otherwise, then the triangle inequality is not satisfied.

- Hierarchical clustering: may combine \(c, e\) first.
Previous clustering algorithms

**Difficulty:** Hard to define the “distance”. E.g., if we assign 1 to \((x, y)\) if \(x\), \(y\) are friends and 0 otherwise, then the triangle inequality is not satisfied

- Hierarchical clustering: may combine \(c, e\) first.
- Assignment-based clustering, e.g., \(k\)-mean: if two random seeds are chosen to be \(b, e\), then \(c\) may be assigned to \(e\)
Previous clustering algorithms

**Difficulty:** Hard to define the “distance”. E.g., if we assign 1 to \((x, y)\) if \(x, y\) are friends and 0 otherwise, then the triangle inequality is not satisfied

- Hierarchical clustering: may combine \(c, e\) first.
- Assignment-based clustering, e.g., \(k\)-mean: if two random seeds are chosen to be \(b, e\), then \(c\) may be assigned to \(e\)
- Spectral clustering: may work. Try it.
A new “distance” measure in social network: **Betweenness**

\[ \text{Betweenness}(A, B): \text{number of shortest paths that use edge } (A, B). \]
Betweenness

A new “distance” measure in social network: **Betweenness**

*Betweenness*$(A, B)$: number of shortest paths that use edge $(A, B)$.

**Intuition:** If to get between two communities you need to take this edge, its betweenness score will be high. An edge with a high betweenness may be a facilitator edge, not a community edge.
1. For each $v \in V$
   
   (a) Run BFS on $v$ to build a directed acyclic graph (DAG) on the entire graph. Give 1 credit to each node except the root.

   (b) Walk back up the DAG and add a credit to each edge. When paths merge, the credits adds up, and when paths split, the credit splits as well.

2. The final score of each edge is the summation of all credits of the $n$ runs, divided by 2.

3. Remove edges with high betweenness, and the remaining connected components are communities.

   **Example** (on board)
Girvan-Newman Algorithm

1. For each $v \in V$
   
   (a) Run BFS on $v$ to build a directed acyclic graph (DAG) on the entire graph. Give 1 credit to each node except the root.

   (b) Walk back up the DAG and add a credit to each edge. When paths merge, the credits add up, and when paths split, the credit splits as well.

2. The final score of each edge is the summation of all credits of the $n$ runs, divided by 2.

3. Remove edges with high betweenness, and the remaining connected components are communities.

**Example (on board)**

What’s the running time?
Can we speed it?
What do the real graphs look like?
**Power-law distributions** A nonnegative random variable $X$ is said to have a power law distribution if

$$\Pr[X = x] \sim cx^{-\alpha},$$

for constants $c > 0$ and $\alpha > 0$.

- **Zipf’s Law**: states that the frequency of the $j$th most common word in English (or other common languages) is proportional to $j^{-1}$

- A city grows in proportion to its current size as a result of people having children.

- Price studied the network of citations between scientific papers and found that the in degrees (number of times a paper has been cited) have power law distributions.

- WWW, ... the rich get richer, heavy tails
Power-law graphs

Natural Graphs:

Yahoo! Web Graph

Top 1% vertices is adjacent to 53% of the edges!
Barabasi-Albert model

When a new node is added to the network at each time $t \in \mathbb{N}$.

1. With a probability $p \in [0, 1]$, this new node connects to $m$ existing nodes uniformly at random.

2. With a probability $1 - p$, this new node connects to $m$ existing nodes with a probability proportional to the degree of node which it will be connected to.

After some math, we get

$$m(k) \sim k^{-\beta},$$

where $m(k)$ is the number of nodes whose degree is $k$, and $\beta = \frac{3-p}{1-p}$. (on board, for $p = 0$)
Thank you!

Some slides are based on the MMDS book
http://infolab.stanford.edu/~ullman/mmds.html

and Jeff Phillips lecture notes
http://www.cs.utah.edu/~jeffp/teaching/cs5955.html