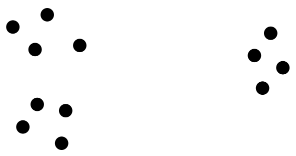
CSE 291: Unsupervised learning **Spring 2008**

Lecture 4 — Hierarchical clustering

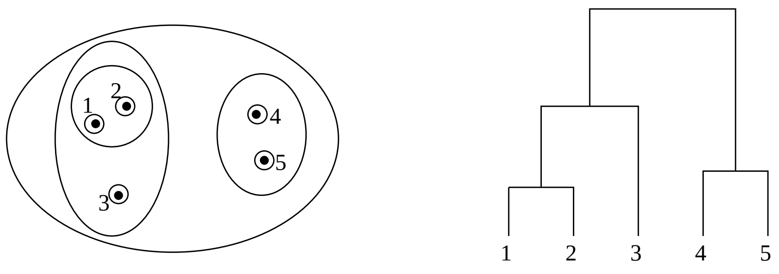
4.1 Multiple levels of granularity

So far we've talked about the k-center, k-means, and k-medoid problems, all of which involve pre-specifying the number of clusters k. How can one do this? Well, if clustering is being used for vector quantization, then k is automatically supplied by the domain and represents the amount of memory available or perhaps the communication bandwidth. On the other hand, if clustering is being used to find meaningful structure in data, then there really is no simple way to know what k ought to be.

In fact, there isn't necessarily a "right" value of k. In the picture below, should k be 2, or 3, or 12?



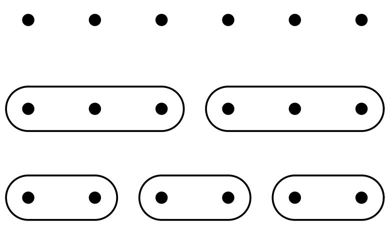
One way to avoid this problem is to do a *hierarchical clustering* of the data. If there are n data points, this is a recursive partitioning into $1, 2, \ldots, n$ clusters. It specifies clusterings at *all* granularities, simultaneously. Here's an example with five data points. The 2-clustering is $\{1,2,3\}, \{4,5\}$, the 3-clustering is $\{1,2\}, \{3\}, \{4,5\}$, and so on. On the right is a *dendrogram* representation of the various clusterings; typically these are drawn so that the height at which a clustering appears is proportional to its cost.



Hierarchical clustering is attractive for obvious reasons. To make matters even better, there are several simple agglomerative algorithms that can be used to construct them; we'll get to these shortly. But are the resulting clusterings any good? Or are they pretty arbitrary?

Let's make this more concrete. Pick any cost function for k-clustering, such as the k-center cost. Is it possible to construct a hierarchical clustering whose induced k-clustering is optimal *for all k*?

The answer is no. To see this, consider the following data set of six points. Below it are shown the optimal 2-clustering and the optimal 3-clustering. These are not nested within each other; therefore they are hierarchically incompatible.



This is bad news. Is it possible that imposing a hierarchical structure forces us to create bad intermediate clusterings? We'll now see that the situation isn't too bad, in that we can always construct a hierarchy that is *close to* optimal for all k.

4.2 Hierarchical clustering with the k-center cost function

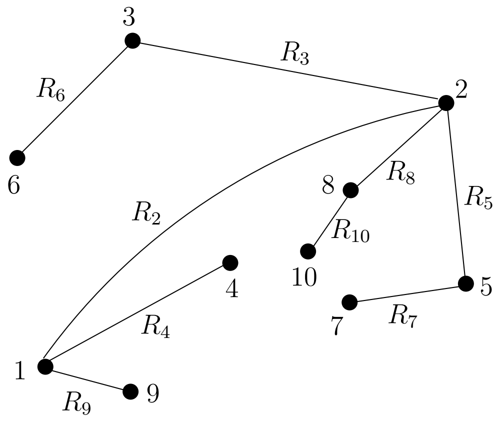
Recall the k-center cost function, set in a metric space $(\mathcal{X}, \rho)$.

> k-CENTER CLUSTERING
>
> *Input:* Finite set $S \subset \mathcal{X}$; integer k.
>
> *Output:* $T \subset X$ with $|T| = k$.
>
> *Goal:* Minimize $\text{cost}(T) = \max_{x \in S} \rho(x, T)$.

Let OPT_k denote the cost of the optimal k-center solution. We'll show how to construct a hierarchical clustering such that for all k, the induced k-clustering has cost $\leq 8\text{OPT}_k$.

4.2.1 Intuition

We saw earlier that a 2-approximate solution to k-center can be obtained by farthest-first traversal. Let's use that traversal to *number* the points (start with any point, then pick the one farthest from it, then the one farthest from those two, and so on). Here's an example.



In this picture, R_i is the distance from the ith point to its closest neighbor in $\{1, 2, \ldots, i-1\}$. So, $\text{cost}(\{1, 2, \ldots, k\}) = R_{k+1}$, and we saw in an earlier lecture that $R_{k+1} \leq 2\text{OPT}_k$. Thus this farthest-first numbering gives us a 2-approximate k-clustering for all k. The hitch is that it is *not* hierarchical. Referring to the picture above, we see that in the 8-clustering, point 10 comes under center 8. And in the 4-clustering, point 8 comes under center 2. Therefore, if we were doing things hierarchically, in the 4-clustering, point 10 would have to come under center 2. But it is much closer to center 4, and the factor-2 guarantee relies on it being assigned to center 4.

First let's figure out how to get a hierarchical clustering. In the picture, i is connected to its closest neighbor $\pi(i) \in \{1, 2, \ldots, i-1\}$, thereby producing a tree structure. We can get a k-clustering by simply dropping the $k-1$ edges $2 \to \pi(2), 3 \to \pi(3), \ldots, k \to \pi(k)$, and treating each connected component as a cluster. These clusterings are indeed nested hierarchically. Let T^{π} denote this hierarchical clustering.

The problem, however, is that the approximation ratio of these induced clusterings could be very bad. Instead, we set up the tree edges a bit differently. We connect each i to a point $\pi'(i) \in \{1, 2 \ldots, i-1\}$ in such a way that the following property holds: any path with increasing node numbers has edge lengths of geometrically decreasing length.

We will build π' by viewing the data at certain specific levels of granularity. Let $R = R_2$; this is some rough measure of the span of the data. If we do not care about distances smaller than R, the entire data set can be summarized by the single point $\{1\}$. This is our coarsest view, and we will call it L_0, granularity level zero. Suppose we want a little more detail, but we still don't care about distances less than $R/2$. Then the data can be summarized by L_0 augmented with $L_1 = \{i : R/2 < R_i \leq R\}$. Continuing in this manner, we construct levels $L_0, L_1, L_2, \ldots$ such that every data point is within distance $R/2^j$ of $L_0 \cup L_1 \cup \cdots \cup L_j$. Now define $\pi'(i)$ to be the closest point to i *at a lower level of granularity.* We will show that $T^{\pi'}$ has provably good induced clusterings.

4.2.2 Algorithm

```
Number the data points by farthest-first traversal
for $i = 2, 3, \ldots, n$:   $R_i \leftarrow \rho(i, \{1, 2, \ldots, i-1\})$
$R \leftarrow R_2$
$L_0 \leftarrow \{1\}$
for $j > 1$:   $L_j \leftarrow \{i : R/2^j < R_i \leq R/2^{j-1}\}$
$\pi'(i) \leftarrow$ closest point to $i$ at lower level of granularity
return $T^{\pi'}$
```

Figure 4.1 shows the effect of this algorithm on our earlier example.

4.2.3 Analysis

Lemma 1. *For any i, $\rho(i, \pi'(i)) \leq R/2^{\text{level}(i)-1}$.*

Proof. Say $i \in L_j$. Let $l+1$ be the lowest-numbered point in L_j. Then $\rho(i, \pi'(i)) \leq R_{l+1} \leq R/2^{j-1}$. □

Lemma 2. *For any k, the induced k-clustering has cost $\leq 4R_{k+1} \leq 8\text{OPT}_k$.*

Proof. Fix any k; the induced k-clustering has centers $1, 2, \ldots, k$.

Pick any i, and follow π'-pointers from i up to some $q \in \{1, 2, \ldots, k\}$. Let q' be the penultimate point on this path, right before q. Then

$$\begin{aligned}
\rho(i, \{1, \ldots, k\}) &\leq \rho(i, q) \\
&\leq \rho(i, \pi'(i)) + \rho(\pi'(i), \pi'(\pi'(i))) + \cdots + \rho(q', q = \pi'(q')) \\
&\leq \frac{R}{2^{\text{level}(i)-1}} + \frac{R}{2^{\text{level}(\pi'(i))-1}} + \cdots + \frac{R}{2^{\text{level}(q')-1}} \\
&\leq \frac{2R}{2^{\text{level}(q')-1}} \leq 4R_{k+1}.
\end{aligned}$$

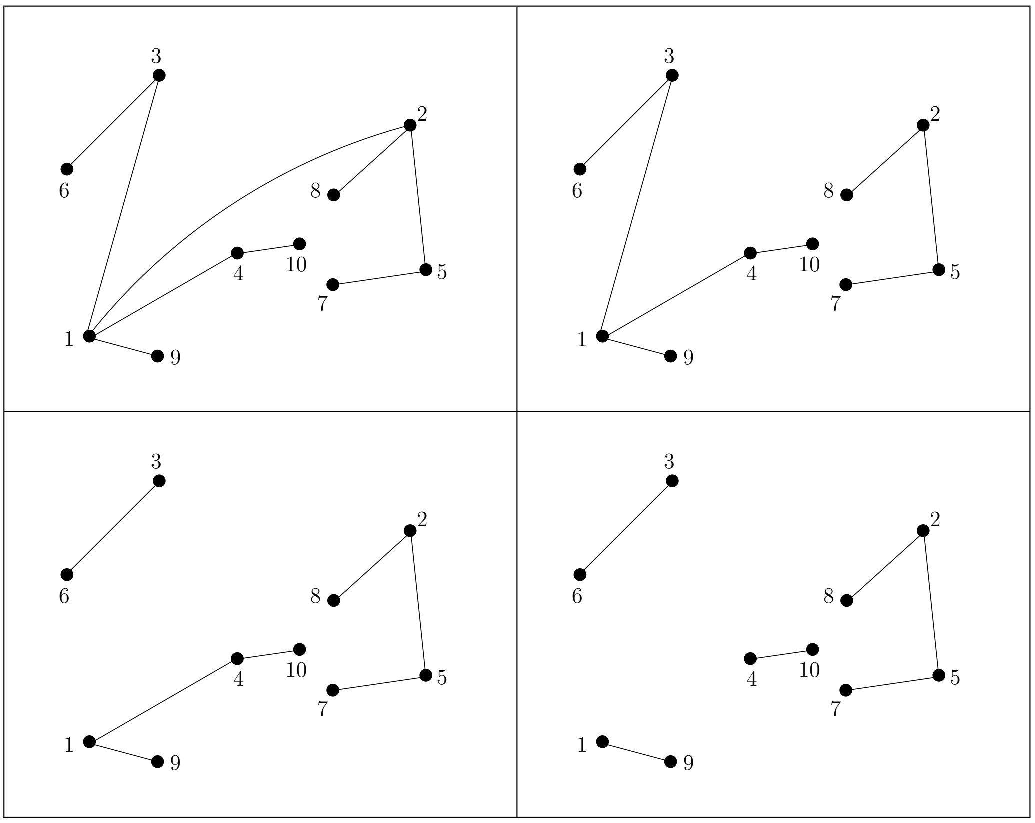


Figure 4.1. A continuation of our earlier example. Shown are the 1-, 2-, 3-, and 4-clusterings obtained from the modified parent function π'.

For the last inequality: the level of $k+1$ is at most that of q', so $R_{k+1} > R/2^{\text{level}(q')}$. □

Problem 1. Find a better approximation algorithm for the hierarchical k-center problem.

Problem 2. Show lower bounds on the achievable approximation factor in hierarchical k-center. This might be a combination of a complexity-based lower bound (hardness of approximating k-center) and a structure-based lower bound (regardless of complexity, the hierarchical structure induces a certain amount of slack).

4.3 Agglomerative algorithms for hierarchical clustering

The most popular algorithms for hierarchical clustering are agglomerative schemes:

```
Start with $n$ clusters, each a single data point.
Repeat:
   Merge the two ‘‘closest’’ clusters.
```

Crucial is the notion of distance between two clusters C, C'. Here are the most common choices:

1. *Single linkage:* $\min\{\rho(i,j) : i \in C, j \in C'\}$

2. *Complete linkage:* $\max\{\rho(i,j) : i \in C, j \in C'\}$

3. *Average linkage.* This comes in three varieties. The last two require data in Euclidean space.

   (a) $\text{mean}\{\rho(i,j) : i \in C, j \in C'\}$

   (b) $\|\text{mean}(C) - \text{mean}(C')\|^2$

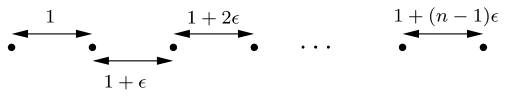
   (c) (Ward's method) $\frac{|C|\cdot|C'|}{|C|+|C'|}\|\text{mean}(C) - \text{mean}(C')\|^2$

4.3.1 Single linkage

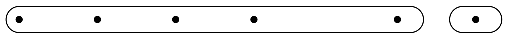
Imagine a process in which a "distance knob" is slowly turned up from zero to infinity. Initially, the n data points are not connected. When the knob passes r, edges are placed between points at distance r from each other. The clustering at any point in time is specified by the connected components of the graph at that time. These are exactly the clusterings induced by single linkage.

This is equivalent to running Kruskal's algorithm on the data points, using interpoint distances as edge weights. To get a k-clustering, drop the $k-1$ largest edges in the minimum spanning tree and output the resulting connected components.

Single linkage has a variety of attractive properties in theory, some of which we will see quite shortly. But in practice it has a disastrous tendency to form long chains. Consider the following set of n points on a line, with distances as shown.



Here ϵ is some miniscule constant. Then the single linkage 2-clustering will have a cluster with $n-1$ points, and another cluster with one point!



4.3.2 Complete linkage

Complete linkage can be thought of as a greedy agglomerative heuristic for hierarchical k-center: it always does the merge that results in the smallest maximum diameter. It is known that its induced k-clusterings are not necessarily close to the optimal k-clusterings in terms of k-center cost: there are examples in which the gap is a multiplicative factor of $k/2$ (for $k \approx \log n$). However, this gap can be shown to be at most $k^{\log_2 3}$.

Problem 3. Obtain matching upper and lower bounds for the approximation factor of complete linkage.

4.3.3 Ward's method of average linkage

Ward's method always does the merge that leads to the smallest increase in k-means cost. To see this, start by fixing any data set $S \subset \mathbb{R}^d$, and recall the k-means cost function: for $T \subset \mathbb{R}^d$,

$$\text{cost}(T) = \sum_{x\in S} \min_{t\in T} \|x-t\|^2.$$

Lemma 3. *For any sets $C, C' \subset \mathbb{R}^d$,*

$$\text{cost}(C \cup C') = \text{cost}(C) + \text{cost}(C') + \frac{|C|\cdot|C'|}{|C|+|C'|}\|\text{mean}(C) - \text{mean}(C')\|^2.$$

Proof. Let $\mu = \text{mean}(C)$, $\mu' = \text{mean}(C')$, and $\overline{\mu} = \text{mean}(C \cup C')$.

$$\begin{aligned}
\text{cost}(C \cup C') - \text{cost}(C) - \text{cost}(C') &= \sum_{x \in C \cup C'} \|x - \overline{\mu}\|^2 - \sum_{x \in C} \|x - \mu\|^2 - \sum_{x \in C'} \|x - \mu'\|^2 \\
&= \sum_{x \in C} \left(\|x - \overline{\mu}\|^2 - \|x - \mu\|^2\right) + \sum_{x \in C'} \left(\|x - \overline{\mu}\|^2 - \|x - \mu'\|^2\right) \\
&= |C| \cdot \|\mu - \overline{\mu}\|^2 + |C'| \cdot \|\mu' - \overline{\mu}\|^2 \\
&= |C| \cdot \left\| \frac{|C'|}{|C| + |C'|}(\mu' - \mu) \right\|^2 + |C'| \cdot \left\| \frac{|C|}{|C| + |C'|}(\mu - \mu') \right\|^2 \\
&= \frac{|C| \cdot |C'|}{|C| + |C'|} \|\mu - \mu'\|^2.
\end{aligned}$$

The third equality uses a special property of squared Euclidean distances that we saw in the lecture on k-means. The fourth equality uses $\overline{\mu} = (|C|\mu + |C'|\mu')/(|C| + |C'|)$. □

Problem 4. How good an approximation to the optimal k-means clustering is the induced k-clustering of Ward's method?

Problem 5. All these agglomerative hierarchical schemes seem to require $O(n^2)$ time, which is prohibitive for massive data sets. Find good ways to speed them up, based on (a) data structures for fast nearest neighbor search, or (b) clever sampling methods.