Chapter 14

Hierarchical Clustering

14.1 Motivation and overview

Given n points in d-dimensional space, the goal of hierarchical clustering is to create a sequence of nested partitions, which can be conveniently visualized via a tree or hierarchy of clusters, also called the cluster *dendogram*. The clusters in the hierarchy range from the fine-grained to the coarse-grained – the lowest level of the tree (the leaves) consists of each point in its own cluster, whereas the highest level (the root) consists of all points in one cluster. Both of these may be considered to be *trivial* clusterings. At some intermediate level, we may find meaningful clusters. As in the case of representative based clustering, if the user supplies k, the desired number of clusters, we can choose the level at which there are k clusters.

More formally, given a dataset $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, where $\mathbf{x}_i \in \mathbb{R}^d$, a clustering $\mathcal{C} = \{C_1, C_2, \cdots, C_m\}$ is a partition of $\mathcal{D}$, i.e., each cluster or member of the partition $C_i \subseteq \mathcal{D}$, such that the clusters are pairwise disjoint $C_i \cap C_j = \emptyset$ (for all $i \neq j$), and $\bigcup_{C_i \in \mathcal{C}} C_i = \mathcal{D}$. A clustering $\mathcal{A} = \{A_1, \cdots, A_r\}$ is said to be nested in another clustering $\mathcal{B} = \{B_1, \cdots, B_s\}$ if and only if $r > s$, and for each cluster $A_i \in \mathcal{A}$, there exists a cluster $B_j \in \mathcal{B}$, such that $A_i \subseteq B_j$. Hierarchical clustering yields a sequence of n nested partitions $\mathcal{C}_1, \cdots, \mathcal{C}_n$, ranging from the trivial clustering $\mathcal{C}_1 = \{\{\mathbf{x}_1\}, \cdots, \{\mathbf{x}_n\}\}$, to the other trivial clustering $\mathcal{C}_n = \{\{\mathbf{x}_1, \cdots, \mathbf{x}_n\}\}$. In general, the clustering $\mathcal{C}_{t-1}$ is nested in the clustering $\mathcal{C}_t$. The cluster dendogram is a rooted binary tree that captures this nesting structure by adding edges between a cluster $C_i \in \mathcal{C}_{t-1}$ and the cluster $C_j \in \mathcal{C}_t$ if C_i is nested in C_j (i.e., if $C_i \subset C_j$). In this way the dendogram captures the entire sequence of nested clusterings.

There are two main algorithmic approaches to mine hierarchical clusters: agglomerative and divisive. Agglomerative strategies work in a bottom-up manner. That is, starting with each of the n points in a separate cluster, they repeatedly merge the most similar pair of clusters until all points are members of the same cluster. Divisive strategies do just the opposite, working in a top-down manner. Starting with all the points in the same cluster, they recursively split the clusters until all

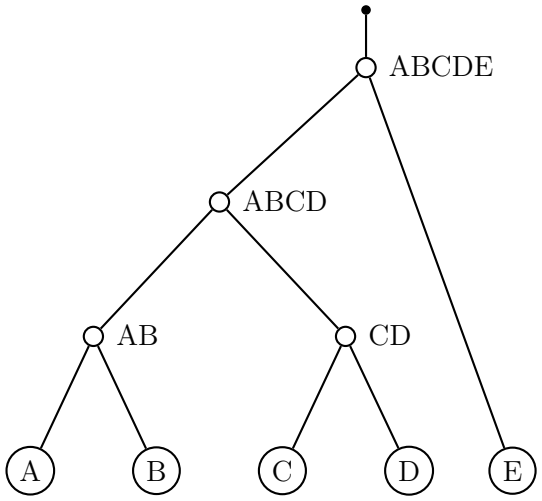

Figure 14.1: Hierarchical Clustering Dendogram

points are in separate clusters.

Example 14.1: Figure 14.1 shows an example of hierarchical clustering of five labeled points, namely A, B, C, D, and E. The dendogram represents the following sequence of nested partitions:

sequence	clusters
$\mathcal{C}_1$	A, B, C, D, E
$\mathcal{C}_2$	AB, C, D, E
$\mathcal{C}_3$	AB, CD, E
$\mathcal{C}_4$	ABCD, E
$\mathcal{C}_5$	ABCDE

Note that each of these clusterings can be obtained by some horizontal cut of the dendogram.

Number of Hierarchical Clusterings

The number of different nested or hierarchical clusterings corresponds to the number of different binary rooted trees with n leaves, with each of the leaves having a different (explicit) label, and each of the internal nodes having an implicit label composed of all the leaf labels below the node. We call such a tree a *dendogram*.

Any tree with t nodes has $t - 1$ edges. Also, any rooted binary tree with m leaves has $m - 1$ internal nodes. Thus, a dendogram with m leaf nodes has a total

of $t = m + m - 1 = 2m - 1$ nodes, and consequently $t - 1 = 2m - 2$ edges. To count the number of different dendogram topologies, let us consider how we can extend a dendogram with m leaves by adding an extra leaf with a new label, to yield a dendogram with $m + 1$ leaves. Note that we can add the extra leaf by splitting (i.e., branching from) any of the $2m - 2$ edges. Furthermore, we can also add the new leaf as a child of a new root, giving $2m - 2 + 1 = 2m - 1$ new dendograms with $m + 1$ leaves. The total number of different dendograms with n leaves is thus obtained by the following product

$$\prod_{m=1}^{n-1}(2m-1) = 1 \times 3 \times 5 \times 7 \times \cdots \times (2n-3) = (2n-3)!! \tag{14.1}$$

The summation above goes up to $n - 1$, since the last term in the product denotes the number of dendograms one obtains when we extend a dendogram with $n - 1$ leaves by adding one more leaf, to yield dendograms with n leaves.

The number of possible hierarchical clusterings is thus given as $(2n - 3)!!$, which grows extremely rapidly. It is obvious that a naive approach of enumerating all possible hierarchical clusterings is simply infeasible.

14.2 Agglomerative Hierarchical Clustering

In agglomerative hierarchical clustering, we begin with each of the n points in a separate cluster. We repeatedly merge the two closest clusters until all points are members of the same cluster, as shown in the pseudo-code given in Algorithm 14.1. Formally, given a set of clusters $\mathcal{C} = \{C_1, C_2, .., C_m\}$, we find the *closest* pair of clusters C_i and C_j and merge them into a new cluster $C_{ij} = C_i \cup C_j$. Next, we update the set of clusters as follows $\mathcal{C} = (\mathcal{C} - C_i - C_j) \cup C_{ij}$. We repeat the process until $\mathcal{C}$ contains only one cluster, or if specified, we stop when there are at most k clusters remaining.

Algorithm 14.1: Agglomerative Hierarchical Clustering Algorithm

AGGLOMERATIVECLUSTERING($\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n, k$):
1 $\mathcal{C} = \{C_i = \{\mathbf{x}_i\} \mid \mathbf{x}_i \in \mathcal{D}\}$ `// Each point in separate cluster`
2 $\mathbf{\Delta} = \{\delta(\mathbf{x}_i, \mathbf{x}_j) : \mathbf{x}_i, \mathbf{x}_j \in \mathcal{D}\}$ `// Compute distance matrix`
3 **while** $|\mathcal{C}| > k$ **do**
4 Find the closest pair of clusters $C_i, C_j \in \mathcal{C}$
5 $C_{ij} = C_i \cup C_j$ `// Merge the clusters`
6 $\mathcal{C} = \{\mathcal{C} - C_i - C_j\} \cup C_{ij}$ `// Update the clustering`
7 Update distance matrix $\mathbf{\Delta}$ to reflect new clustering

The main step in the algorithm is how to define the closest pair of clusters. Several distance measures, such as single link, complete link, group average, and others

discussed below, can be used to compute the distance between any two clusters. The between cluster distances are ultimately based on the distance between two points, which is typically given by the Euclidean distance:

$$\delta(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}$$

However, one may use any other distance metric, or if available, one may use an application specific distance matrix between pairs of points.

Single Link

Given two clusters C_i and C_j, the distance between them, denoted $\delta(C_i, C_j)$ is defined as the minimum distance between a point in C_i and a point in C_j:

$$\delta(C_i, C_j) = \min\{\delta(\mathbf{x}, \mathbf{y}) \mid \mathbf{x} \in C_i, \mathbf{y} \in C_j\} \tag{14.2}$$

The name, single link, comes from the observation that if we choose the minimum distance between points in the two clusters, and connect those points, then (typically) only a single link would exist between those clusters, since all other pairs of points would be farther away.

Complete Link

The distance between two clusters is defined as the maximum distance between a point in C_i and a point in C_j:

$$\delta(C_i, C_j) = \max\{\delta(\mathbf{x}, \mathbf{y}) \mid \mathbf{x} \in C_i, \mathbf{y} \in C_j\} \tag{14.3}$$

The name, complete link, conveys the fact that if we connect all pairs of points from the two clusters with distance at most $\delta(C_i, C_j)$, then all possible pairs would be connected, i.e., we get a complete linkage.

Group Average

The distance between two clusters is defined as the average pairwise distance between points in C_i and C_j:

$$\delta(C_i, C_j) = \frac{\sum_{\mathbf{x} \in C_i} \sum_{\mathbf{y} \in C_j} \delta(\mathbf{x}, \mathbf{y})}{n_i \cdot n_j} \tag{14.4}$$

where $n_i = |C_i|$ denotes the number of points in cluster C_i.

Mean Distance

The distance between two clusters is defined as the distance between the means or centroids of the two clusters:

$$\delta(C_i, C_j) = \delta(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j) \tag{14.5}$$

where $\boldsymbol{\mu}_i = \frac{1}{n_i}\sum_{\mathbf{x}\in C_i} \mathbf{x}$.

Minimum Variance: Ward's Method

The distance between two clusters is defined as the increase in the sum of squared errors (SSE) when the two clusters are merged. The SSE for a given cluster C_i is given as

$$SSE_i = \sum_{\mathbf{x}\in C_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 \tag{14.6}$$

which can also be written as

$$\begin{aligned} SSE_i &= \sum_{\mathbf{x}\in C_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 \\ &= \sum_{\mathbf{x}\in C_i} \mathbf{x}^T\mathbf{x} - 2\sum_{\mathbf{x}\in C_i} \mathbf{x}^T\boldsymbol{\mu}_i + \sum_{\mathbf{x}\in C_i} \boldsymbol{\mu}_i^T\boldsymbol{\mu}_i \\ &= \sum_{\mathbf{x}\in C_i} \mathbf{x}^T\mathbf{x} - n_i\boldsymbol{\mu}_i^T\boldsymbol{\mu}_i \end{aligned} \tag{14.7}$$

The SSE for a clustering $\mathcal{C} = \{C_1, \cdots, C_m\}$, is given as

$$SSE = \sum_{i=1}^{m} SSE_i = \sum_{i=1}^{m}\sum_{\mathbf{x}\in C_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 \tag{14.8}$$

Ward's measure defines the distance between two clusters C_i and C_j, as the net change in the SSE value when we merge C_i and C_j into C_{ij}, given as

$$\delta(C_i, C_j) = \Delta SSE_{ij} = SSE_{ij} - SSE_i - SSE_j \tag{14.9}$$

We can obtain a simpler expression for the Ward's measure by plugging (14.7) into (14.9), and noting that, since $C_{ij} = C_i \cup C_j$, and $C_i \cap C_j = \emptyset$, we have $|C_{ij}| = n_{ij} = n_i + n_j$.

$$\begin{aligned} \delta(C_i, C_j) &= \Delta SSE_{ij} \\ &= \sum_{\mathbf{z}\in C_{ij}} \|\mathbf{z} - \boldsymbol{\mu}_{ij}\|^2 - \sum_{\mathbf{x}\in C_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 - \sum_{\mathbf{y}\in C_j} \|\mathbf{y} - \boldsymbol{\mu}_j\|^2 \\ &= \sum_{\mathbf{z}\in C_{ij}} \mathbf{z}^T\mathbf{z} - n_{ij}\boldsymbol{\mu}_{ij}^T\boldsymbol{\mu}_{ij} - \sum_{\mathbf{x}\in C_i} \mathbf{x}^T\mathbf{x} + n_i\boldsymbol{\mu}_i^T\boldsymbol{\mu}_i - \sum_{\mathbf{y}\in C_j} \mathbf{y}^T\mathbf{y} + n_j\boldsymbol{\mu}_j^T\boldsymbol{\mu}_j \\ &= n_i\boldsymbol{\mu}_i^T\boldsymbol{\mu}_i + n_j\boldsymbol{\mu}_j^T\boldsymbol{\mu}_j - (n_i + n_j)\boldsymbol{\mu}_{ij}^T\boldsymbol{\mu}_{ij} \end{aligned} \tag{14.10}$$

The last step follows from the fact that $\sum_{\mathbf{z}\in C_{ij}} \mathbf{z}^T\mathbf{z} = \sum_{\mathbf{x}\in C_i} \mathbf{x}^T\mathbf{x} + \sum_{\mathbf{y}\in C_j} \mathbf{y}^T\mathbf{y}$. Noting that

$$\boldsymbol{\mu}_{ij} = \frac{n_i\boldsymbol{\mu}_i + n_j\boldsymbol{\mu}_j}{n_i + n_j}$$

we obtain

$$\boldsymbol{\mu}_{ij}^T\boldsymbol{\mu}_{ij} = \frac{1}{(n_i+n_j)^2}\left(n_i^2\boldsymbol{\mu}_i^T\boldsymbol{\mu}_i + 2n_in_j\boldsymbol{\mu}_i^T\boldsymbol{\mu}_j + n_j^2\boldsymbol{\mu}_j^T\boldsymbol{\mu}_j\right)$$

Plugging the above into (14.10), we have

$$\begin{aligned}
\delta(C_i, C_j) &= \Delta SSE_{ij} \\
&= n_i\boldsymbol{\mu}_i^T\boldsymbol{\mu}_i + n_j\boldsymbol{\mu}_j^T\boldsymbol{\mu}_j - \frac{1}{(n_i+n_j)}\left(n_i^2\boldsymbol{\mu}_i^T\boldsymbol{\mu}_i + 2n_in_j\boldsymbol{\mu}_i^T\boldsymbol{\mu}_j + n_j^2\boldsymbol{\mu}_j^T\boldsymbol{\mu}_j\right) \\
&= \frac{n_i(n_i+n_j)\boldsymbol{\mu}_i^T\boldsymbol{\mu}_i + n_j(n_i+n_j)\boldsymbol{\mu}_j^T\boldsymbol{\mu}_j - n_i^2\boldsymbol{\mu}_i^T\boldsymbol{\mu}_i - 2n_in_j\boldsymbol{\mu}_i^T\boldsymbol{\mu}_j - n_j^2\boldsymbol{\mu}_j^T\boldsymbol{\mu}_j}{n_i+n_j} \\
&= \frac{n_in_j\left(\boldsymbol{\mu}_i^T\boldsymbol{\mu}_i - 2\boldsymbol{\mu}_i^T\boldsymbol{\mu}_j + \boldsymbol{\mu}_j^T\boldsymbol{\mu}_j\right)}{n_i+n_j} \\
&= \left(\frac{n_in_j}{n_i+n_j}\right)\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|^2
\end{aligned} \tag{14.11}$$

Ward's measure can be thought of as a weighted version of the mean distance measure, since if we use Euclidean distance, the mean distance measure given in (14.5) can be rewritten as:

$$\delta(\boldsymbol{\mu}_{C_i}, \boldsymbol{\mu}_{C_j}) = \left\|\boldsymbol{\mu}_{C_i} - \boldsymbol{\mu}_{C_j}\right\|^2 \tag{14.12}$$

We can see that the only difference is that the Ward's measure weights the distance between the centroids by half of the harmonic mean of the cluster sizes. Recall that the harmonic mean of two numbers n_1 and n_2 is given as $\frac{2}{\frac{1}{n_1}+\frac{1}{n_2}} = \frac{2n_1n_2}{n_1+n_2}$.

Example 14.2 (Single Link Example): Consider the single link clustering example shown in Figure 14.2, which consists of five points, whose pair-wise distances are also shown on the top left. Initially all points are in their own cluster. The closest pair of points are (A, B) and (C, D), both with $\delta = 1$. We choose A and B to merge, and derive a new distance matrix for the merged cluster. Essentially, we have to compute the distances of the new cluster AB to all other clusters. For example, $\delta(AB, E) = 3$, since $\min\{\delta(A, E) = 4, \delta(B, E) = 3\} = 3$. At the next step, we merge C and D, since they are the closest clusters, and we obtain a new distance matrix for the current set of clusters. After this, AB and CD are merged,

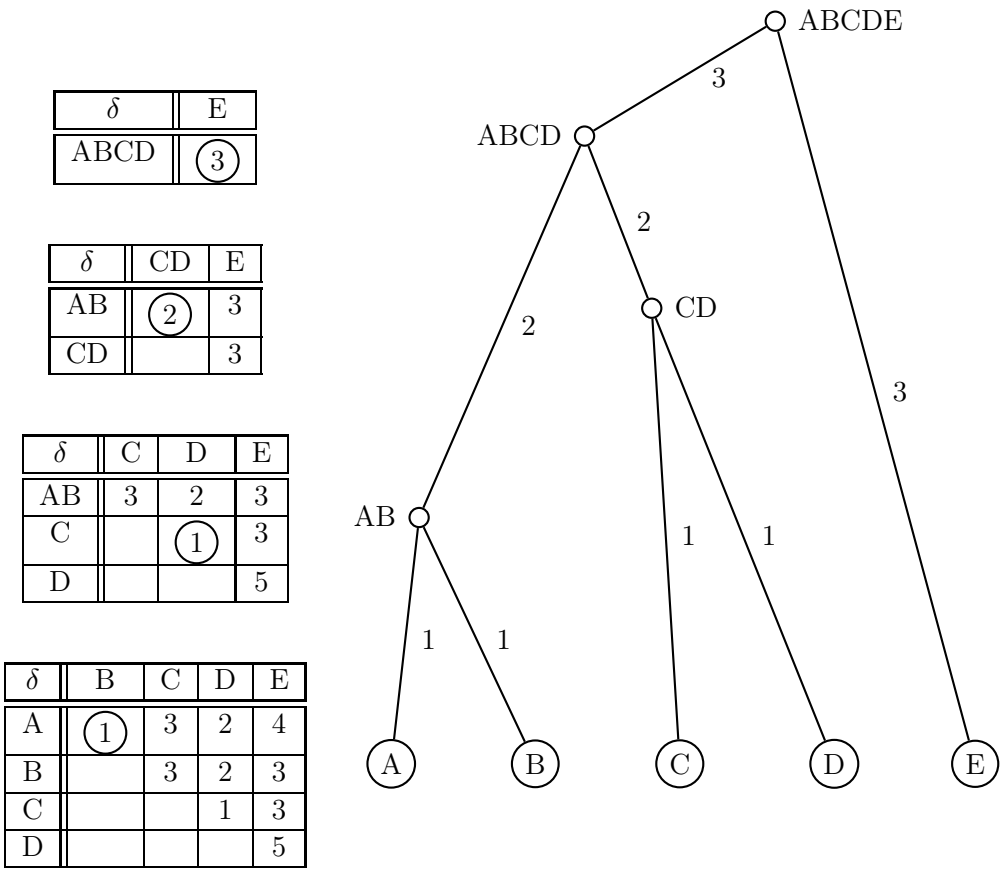

Figure 14.2: Single Link Agglomerative Clustering

and finally, E is merged with $ABCD$. In the distance matrices, we have shown (circled) the minimum distance used at each iteration that results in a merging of two closest pairs of clusters.

Graph-based Interpretation

Single- and Complete-link methods also have a graph-based interpretation. If we treat each of the points as a vertex, and add edges between two nodes with distance less than some threshold value, then the connected components in the graph correspond to set of single-link clusters using that threshold. For example, if we were to choose $\delta = 1$, then the connected components, shown in Figure 14.3a correspond to the clusters in Figure 14.2 with merging threshold $\delta = 1$. The same holds when we use $\delta = 2$ as shown in Figure 14.3b. On the other hand, given that a graph contains edges only between nodes with distances less than a given threshold, then the set of

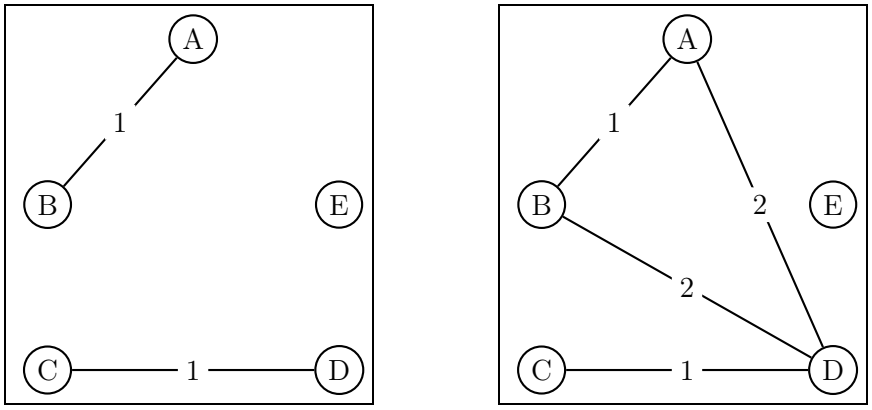

Figure 14.3: Single Link Clustering and Connected Components: (a) $\delta = 1$, clusters are AB, CD, E. (b) $\delta = 2$, clusters are $ABCD$, E.

maximal cliques in the graph corresponds to the set of complete-link clusters at that threshold.

Lance-Williams Formula for Cluster Proximity

Whenever two clusters C_i and C_j are merged into C_{ij}, we need to recompute the distances from the newly created cluster C_{ij} to all other clusters C_k ($k \neq i$ and $k \neq j$). The Lance-Williams formula provides a general equation to recompute the distances for several of the commonly used cluster proximity measures proposed above:

$$\delta(C_{ij}, C_k) = \alpha_i \cdot \delta(C_i, C_k) + \alpha_j \cdot \delta(C_j, C_k) + \beta \cdot \delta(C_i, C_j) + \gamma \cdot |\delta(C_i, C_k) - \delta(C_j, C_k)| \tag{14.13}$$

The weight terms $\alpha_i, \alpha_j, \beta, \gamma$ differ from one measure to another. Let $n_x = |C_x|$ denote the cardinality of cluster C_x, then the weight terms for the different distance measures are given as follows:

Method	α_i	α_j	β	γ
Single Link	$\frac{1}{2}$	$\frac{1}{2}$	0	$-\frac{1}{2}$
Complete Link	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$
Group Average	$\frac{n_i}{n_i+n_j}$	$\frac{n_j}{n_i+n_j}$	0	0
Mean Distance	$\frac{n_i}{n_i+n_j}$	$\frac{n_j}{n_i+n_j}$	$\frac{-n_i \cdot n_j}{(n_i+n_j)^2}$	0
Ward's	$\frac{n_i+n_k}{n_i+n_j+n_k}$	$\frac{n_j+n_k}{n_i+n_j+n_k}$	$\frac{-n_k}{n_i+n_j+n_k}$	0

Computational Complexity

In agglomerative clustering, we need to compute the distance of each cluster to all other clusters, and at each step the number of clusters decreases by one. Initially it takes $O(n^2)$ time to create the pairwise distance matrix, unless it is specified by the application.

At each merge step, the distances from the merged cluster to the other clusters have to be recomputed, whereas the distances between the other clusters remain the same. This means that in step t, we compute $O(n - t)$ distances. The other main operation is to find the closest pair in the distance matrix. For this we can keep the $O(n^2)$ distances in a heap data structure, which allows us to find the minimum distance in $O(1)$ time. However, creating the heap takes $O(n^2)$ time (to insert the n^2 pairwise distances), and deleting/updating $O(n - t)$ distances from the merged cluster takes $O(n \log n)$ time. The total time across all iterations is $O(n^2 \log n)$, since there are n steps and each step takes $O(n \log n)$ time. Also note that all need $O(n^2)$ space to store the distance matrix or the heap.

14.3 Distances on Dendograms: Tree Metrics

Add algos from bioinfo

14.4 Annotated References