A Brief Guide to Cluster Analysis
Oliver Hatfield
Supervisors: David Leslie, Nicos Pavlidis

What is Cluster Analysis?
When analysing data, one often wants to organise it into clusters of similar data. Each cluster will consist of data points which are in some sense similar or close, but far away from other data. Some example applications include:
- classifying animals in biology (taxonomy).
- grouping customers for targeted advertising.
- image recognition software.

The Problem: General Setting
In a general clustering problem, the following setup is typical:
- There exists a data set, \( X = \{x_i\}_{i=1}^n \).
- The aim is to assign a label, \( z_i \), to each data point \( x_i \). Each label \( z_i \in c_1 \ldots c_k \), where each \( c_j \) corresponds to cluster \( j \).
- Labels should be assigned such that ‘similar’ points have the same label, but ‘dissimilar’ points should be labelled differently.
- The number of clusters, \( m \), can be fixed or unknown.

One major complication in cluster analysis is that the concepts of ‘similar’ and ‘close’ are not well defined, and varies according to context.

Figure 1 shows an example of a set of data to be clustered. Even though these clusters are obvious to humans, (Jain, 10) claims no current clustering algorithm can detect all of these. This is largely due to the complex nature of some of the shapes. This poster therefore considers examples of clustering algorithms, and examines their advantages.

Early Methods: K-Means Clustering
One of the first clustering algorithms was K-means clustering. (Figure 2).
- Select a number of clusters to detect, \( K \), and assign each \( x_i \) to a cluster \( c_k \).
- Calculate the mean of each cluster, \( \mu_k = \frac{1}{n_k} \sum x_i \) where \( n_k \) is the number of data points in cluster \( k \).
- Reassign each \( x_i \) to the cluster corresponding to the closest \( \mu_k \).
- Repeat the previous two steps until no change is seen.

Figure 2: K-means clustering. Initially, each point \( x_i \) is randomly assigned to a cluster. At each step, each \( x_i \) is reassigned to the cluster of the nearest mean (bold triangles), before means are re-calculated (empty triangles).

[+] Computationally quick, and straightforward to implement.
[-] Needs specification of how many clusters, \( K \), to find.
[-] Minimises distances to cluster centres, \( \Rightarrow \) finds ‘circular’ clusters.

Alternative Method: Spectral Clustering
Here, the idea is to create a ‘similarity matrix’ of the data, and then do K-means clustering on its eigenvectors. Each entry, \( s_{ij} \) of the similarity matrix, is the similarity between points \( x_i \) and \( x_j \). Example similarity functions are:
- \( \text{Gaussian pdf of the Euclidean distances for selected parameter } \sigma \).
- \( \epsilon \)-neighbourhood graph: \( s_{ij} = 1/(d(x_i,x_j) < \epsilon) \). So \( s_{ij} = 1 \) for close points, or 0.
- Finally, \( d_i = \sum s_{ij} \), and \( D \) is a matrix with \( d_i \) on the diagonal entries, but 0 everywhere else. A graph Laplacian, \( L \), is then constructed from \( D-W \). A simple example is \( L = D-W \). Spectral clustering then runs as follows.
- Find the first \( K \) eigenvectors, \( w_1 \ldots w_K \), of \( L \).
- Let \( U \) be a matrix with \( w_i \) as columns.
- Perform K-means clustering on the \( n \) rows of \( U \), called \( y \).

The clustering of \( y \) is then identical to the clustering of \( x_i \) on the original data.\n
[+] Better able to detect different shaped clusters. (See Figure 3).
[-] Computationally much slower than K-means: unusable online.
[-] Still needs specification of \( K \).

Figure 3: Spectral clustering easily detects the spirals (left). However the right two images use identical data, and show how randomness in initialisation of the K-means step can cause problems with additional clusters.

Online Clustering
Traditionally, clustering is done at the end of an experiment with a data set, \( X \), as above. However, this is often inappropriate. For example, a company wanting to cluster its sales data into groups of customers can lose an edge in competition by waiting before clustering. They could instead cluster data as it arrives. This poster therefore also considers online clustering with constant parameters. Further work would consider non-stationary parameters.

Online Clustering: Particle Filters
In order to perform online clustering, consider particle filtering. It approximates the posterior distribution of the cluster labels, \( \rho(z_{1:n} | x_{1:n}) \).
- When \( n = 1 \), \( \rho(z_1 = 1) \) the only data point is in its own cluster.
- Thereafter, seek to approximate the posterior with a discrete distribution by using \( N \) particles, \( \{z_{1:n}^{(i)}\}_{i=1}^N \). Each has weight \( w_{n}^{(i)} \), sum to 1.
- Therefore \( E(g(z_{1:n})) \approx \sum_{i=1}^{N} w_{n}^{(i)} g(z_{1:n}^{(i)}) \) for any function \( g(z_{1:n}) \).

Given \( \{z_{1:n}^{(i)}\}_{i=1}^N \) and a new observation \( x_{n+1} \), each of the next \( N \) particles \( z_{n+1}^{(i)} \) is a sample from the posterior \( \rho(z_{1:n+1} | x_{1:n+1}) \). This sampling is basically done using importance sampling, with the prior \( \rho(z_{1:n+1} | x_{1:n+1}) \) as a proposal density. The following procedure is similar, but more efficient.

1. For a particle \( z_{1:n}^{(i)} \) with \( k_i \) distinct clusters, there are \( k_i + 1 \) different choices for cluster of \( x_{n+1} \): each of the \( k_i \) existing clusters, or a new one. For each \( z_{1:n}^{(i)} \) and \( j = 1 \ldots k_i + 1 \), the possible \( z_{n+1}^{(i)} = \text{th} \) of the form \( (z_{1:n}^{(i)}, j) \). Let \( w_{n+1}^{(i)} \propto \rho(z_{1:n}^{(i)}, x_{n+1}) / \rho(z_{1:n}^{(i)} | x_{n+1}) \) normalised to sum to 1.
2. Since this creates many particles at each time step, one must get rid of all but \( N \) particles each time to reduce the problem. But how?
   - Retain the most likely possible particles (where \( w_{n+1}^{(i)} > 1/c \), for some c).
   - Sample randomly from the remaining particles.
   - Change the sampled particles’ weights to \( 1/c \).

Finally, given particles \( \{z_{1:n+1}^{(i)}\}_{i=1}^N \), Bayes’ Thm. shows these help estimate:
\[
\rho(z_{1:n+1} | x_{1:n+1}) \propto \sum_{i=1}^{N} w_{n}^{(i)} \rho(z_{1:n+1} | z_{1:n}^{(i)}) \rho(z_{1:n}^{(i)} | x_{1:n+1}) \tag{1}
\]
To conclude, cluster the data by selecting the \( z_{n+1}^{(i)} \) which maximises this.

Figure 4: A particle filter clustering. Black stars represent a ‘noise’ cluster. (Figure from Schubert, 2005).

Conclusion
Many different clustering algorithms exist, each designed to solve an important data analysis problem. While there is still room for improvement, significant advancements have been made in this field.

- K-means clustering is simple, fast, and effective for simple problems.
- Spectral clustering is more advanced, but is slower.
- Of these, particle filters are most efficient for online clustering.

http://www.lancaster.ac.uk/pg/hatfield/
o.hatfield1@lancaster.ac.uk