2.9 Microstate Clustering of MD Conformations

Although there is a lack of consensus about the best method to cluster kinetically related conformations, the most usual methods are based on some sort of geometrical clustering. The assumption behind structural clustering is that structures closely related in the geometrical space should also be closely related in the kinetic space, hence, grouping structures that are close in geometry will approximately give structures that are close in dynamics. Several structure-based clustering methods are already available, with the most fundamental and widely used are: K-centers, K-means and K-medoids clustering. The common goal for these three methods is to partition a set of n conformations into k mutually exclusive partitions C_1, $C_2, \ldots, C_k$. These k partitions are then used for macrostate lumping in the later stage.

K-centers clustering aims at find k "centers" (see Fig. 2.4A), which is defined by a subset S from the set of points V such that $|S| = k$ and minimizing the expression:

$$\max_{v \in V} \min_{s \in S} (v, s) \tag{2.5}$$

or, in simple words, find k points from the dataset such that the longest distance between any point to its closest corresponding center is minimized. The k partitions can then be obtained by assigning all points into their closest corresponding centers to form k mutually exclusive groups.

The k-centers problem is actually NP-hard, which implies that solving the exact solution is computationally expensive. In real practice though, k-centers clustering algorithm usually refers to an approximate algorithm shown below:

1. Randomly select one conformation as the center of the first microstate k_1.
2. Calculate the distance $d(x_i, k_1)$ between each of the conformations x_i in the dataset and k_1.
3. Choose the conformation with the largest $d(x_i, k_1)$ value as the second microstate center k_2.
4. Reassign the conformations in the dataset to the new cluster if the distance to the new cluster center is shorter than the distance to any other cluster centers (i.e. for a new cluster center k_2, conformation x_i is assigned to C_2 if $d(x_i, k_2)$ is shorter than $d(x_i, k_1)$.
5. Then choose the next cluster center that is furthest from the all previous centers and repeat step 4.
6. Repeat the same procedure until the desired number of microstates is obtained.

The k-centers clustering method can create clusters with an approximately equal geometric volume. Moreover, the clustering speed can be greatly improved by applying triangle inequality in the step of cluster assignment, which has been currently implemented in the MSMBuilder package [34, 50, 51].

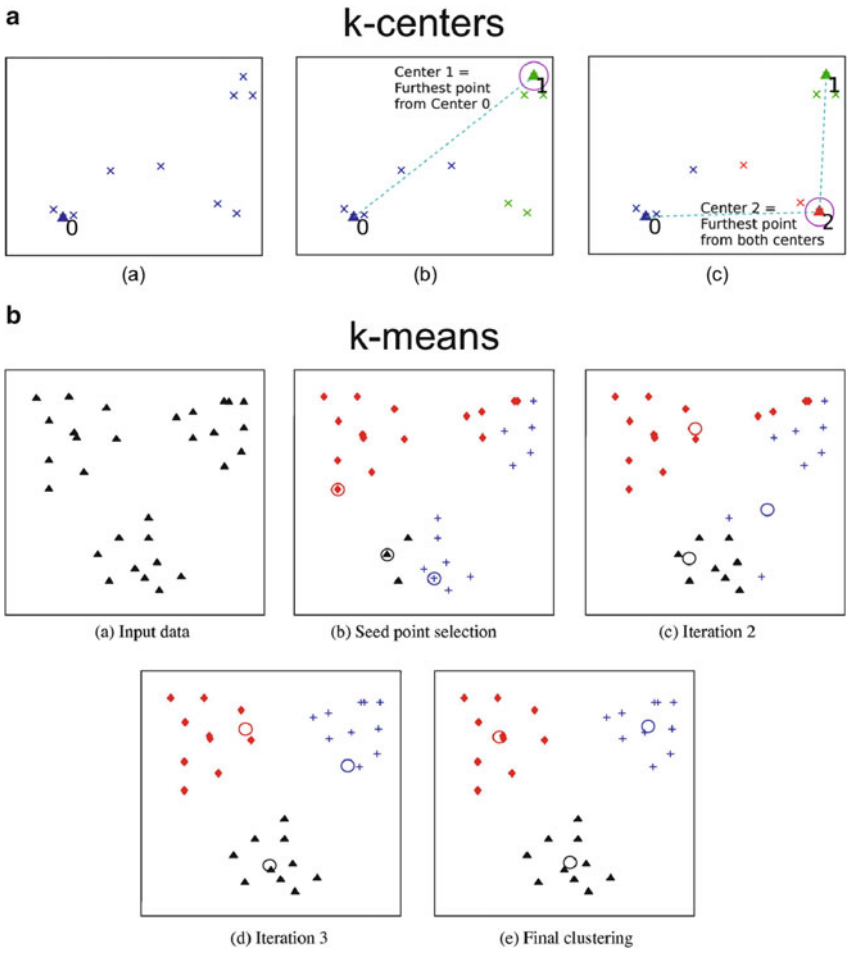

Fig. 2.4 (**A**) **Illustration of an approximate k-centers clustering algorithm.** The process of generating k geometric groups from a given dataset with the approximate k-centers algorithm is illustrated as follows: (from *left* to *right*) (*a*) From the given data points, choose a random point as the first cluster center. (*b*) Measure the distance of all points against the first center and choose the one with the furthest distance as the second cluster center. Assign all points to their closest cluster center such that all points are divided into two clusters ("partitions" in the mathematical sense), illustrated here with two different colors. (*c*) Measure the distance of all points against their assigned cluster centers, find the point with the maximum distance (i.e. furthest from all existing center) as the next cluster center. Re-assign all points to their closest centers into partitions. Repeat until the desired number of clusters k is obtained. The final partitioning is used as the geometrical grouping of the points (Figure adapted from reference [48]). (**B**) **Illustration of an approximate k-means clustering algorithm.** K-means algorithm attempts to divide the given dataset (*a*) into k geometric partitioning in the following way (From *left* to *right*, *top* to *bottom*). (*b*) From the data points, randomly choose k points as initial centers (*circled*). Assign all points to their closest corresponding centers into k partitions, shown here in different colors. (*c–d*) For each partition, take the "mean" position of the points within the group as the updated center position (*circles*). Re-assign all the points again to the new centers. (*e*) Repeat the process until no change in the cluster assignment is observed. The final cluster assignment is taken as the geometrical partitioning of the points (Figure adapted from reference [49])

K-means clustering refers to something very different from k-centers clustering (see Fig. 2.4B). Instead of aiming solely at points to be centers, the k-means clustering attempts to find k partitions so as to minimize:

$$\sum_{K=1}^{k} \sum_{x_i \in C_k} \sqrt{\left(x_i - \overline{k}_j\right)^T \left(x_i - \overline{k}_j\right)} \tag{2.6}$$

where $\overline{k}_j = \frac{1}{N_k} \sum_{x_i \in C_k} x_i$. In other words, the points are put into k partitions so that the sum of distances of all points to the partition average of their assigned partitions is minimized.

Just like k-centers, k-means is also an NP-hard problem, and so an approximation is also needed. A commonly used approximate k-means clustering protocol is illustrated here:

1. Instead of using one conformation as the first microstate center, k conformations are (randomly) chosen as the initial centers for the k microstates.
2. Calculate the Euclidean distance between every conformations in the dataset to each centers defined in step 1.
3. Assign the conformation to the microstate with the minimum distance.
4. Determine the mean vectors by averaging the distance vector for all the conformations within each microstate, and using the mean vector as the new center.
5. Repeat step 2, 3 and 4 until the clustering process is converged. That is, the new round of iteration does not change the assignments of any conformations from the previous iteration.

Despite the popularity of k-means cluster, this clustering technique is actually sensitive to the low density regions and tends to lump the points from the low density regions into the clusters from high density regions, which in the context of microstate clustering leads to the incorrect description of the some interesting states such as the transition states. A clustering algorithm that closely resembles k-means, which is known as k-medoids clustering, can overcome the above drawbacks of k-means by taking actual data points ("medoids") instead of the means of the partitions as centers. Unfortunately, k-medoids also has its own limitations, such as being inefficient for large data sets and offer a poor control of the cluster size.

2.10 Implied Timescales and Number of Macrostates

With the microstates generated from the first stage of clustering, we can construct the TPM based on the transitions between these states. Then if we attempt to do eigenvector decomposition of the TPM:

$$X_i P(\tau) = \mu_i X_i \tag{2.7}$$