PA2-Clustering

Problem 1 Clustering synthetic data Problem 2 Image segmentation

翟冠勋 ▸ 54345382 ▸ 4/12/2015

PA2-Clustering

Problem 1 Clustering synthetic data
Problem 2 Image segmentation

Part I
Algorithm Implementation

1. K-means
   K-means is a 0-1 label clustering algorithm. It is based on the Euclid distance from each point to the center of each cluster.
   And afterwards, the center is recalculated according to the label, which is represented by the Zij matrix.

   To implement this algorithm, the initial cluster centers, which are the means of the data in different clusters, are needed to be set.
   Then during the iteration step, some tricks of matlab is needed to calculate the Euclid distance.
   Since K-means is relatively simple comparing with the other 2 algorithm, the iteration can be set to a finite criteria. Here I use isequal() function to determine where to stop iteration, which means the value doesn't change any more.

2. EM-GMM
   EM Algorithm is a continuous label, which means the Zij is not an integral. That is because EM is based on the maximum likelihood estimate. With different estimate, the algorithm will be different. Here we use the GMM as the estimate.
   EM includes 2 iteration steps. The first is expectation, which aims on calculating the probability that point Xi is in each cluster. This step is calculated with the π, the ratio of the points in each cluster, along with the Gaussian distribution of Xi in each cluster.
   After the E-step, each point has a proper distribution on different clusters. Then we use the Zij matrix, along with the Xi to calculate the mean, variance, numbers of points and percentage of points in the total data in each cluster. This is called M-step. Here M represents maximum. And then repeat the E-step and M-step to iterate and get the final clustering result of the data.

   The implementation of the EM is more complex than K-means. Also, the calculation needed is also quite large. Sometimes, if the criteria is too strict, the iteration may not stop. And we need to set a finite iteration times to end the algorithm. Also, the result may not be correct without proper implementation. The coding part does not have much trick. We just need to follow the instructions of the textbook and assignment document. However, the

result may not be good, which is happened in my program. Unfortunately, I didn't find a good solution the fix the algorithm.

3. Mean-shift
   Mean-shift Algorithm is connected with kernel density estimate. This method uses a kernel with bandwidth of h to set cluster. Here we use the Gaussian Kernel.
   This algorithm also needs a lot of calculation. Because it will have a distribution used on all the data to get the iterated value of 1 point, which means the calculation needed will be a squared value.
   However, the method is not complex, and the parameter needed is also not very much.
   During iteration, the points will converge to several peaks. Then we use these peaks to set labels for all the points.

   The implementation of the iteration is relatively simple. But the cluster choosing is a little bit difficult. Since the data may converge to the same value in different area. So we need all the data and many comparing steps to get the number of the clusters. And this number may not be fixed to the same value. How to find the peaks and assign the label effectively is the main question. My method for this is by setting those points that are already compared and assigned to 0 value. Then find new cluster label and repeat the step. Since matlab is mainly used for matrix calculation, thus some looping steps can be omitted. And because the data has more than 1 dimension, to compare the value of each point with 0, I used multiply and plus of all the data and add the results afterwards to get an identical value for each point. Then I only have to deal with a vector, which is much simpler than a matrix.

Part II

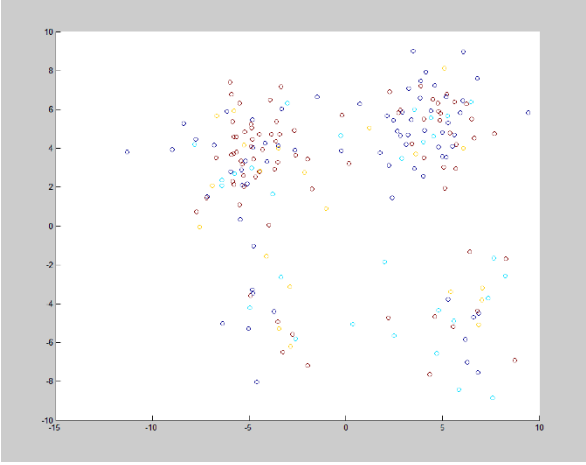
Clustering synthetic data.

The implementation is the question (a) of the 1 problem.

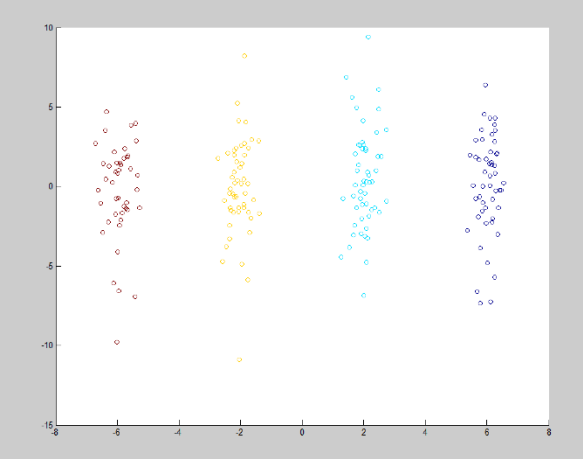
About question (b), there are 3 sets of data, A, B, C. And we need to implement the 3 algorithm written before on these 3 trails.

K-means

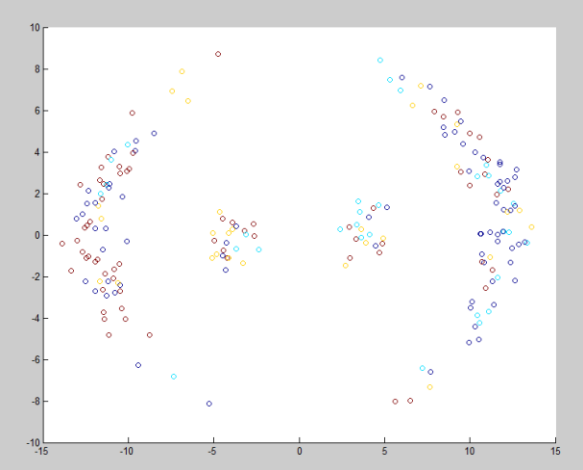
Trail A



Trail B



Trail C



Obviously, using K-means algorithm, the Trail B has the best clustering result. And for the other 2 methods, the result is not quite good. The points all mix together.
That might be due to the separation of the data. We can see than the result in trail A is slightly better than trail C. Even though, since the data distribution of trail A is not very regular, the result is still not very good.

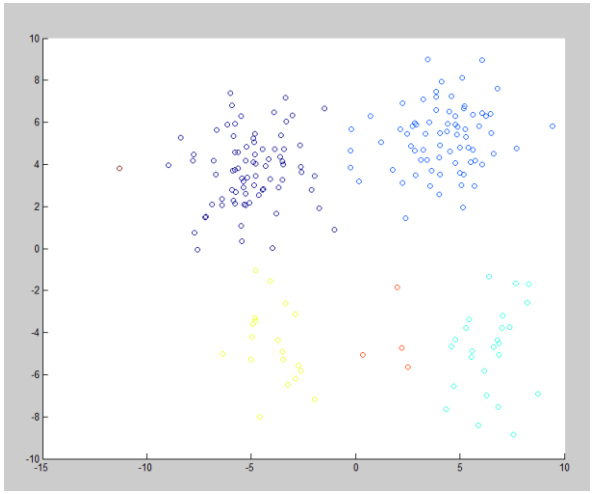
EM-GMM

Unfortunately, my EM program is not correct. The final result of the iteration are all converge to a single cluster.

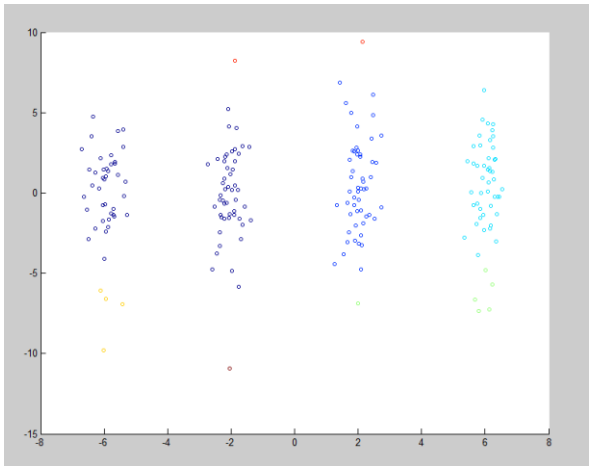
I think that is because the criteria setting. Or just some bugs in the program.

Mean-shift

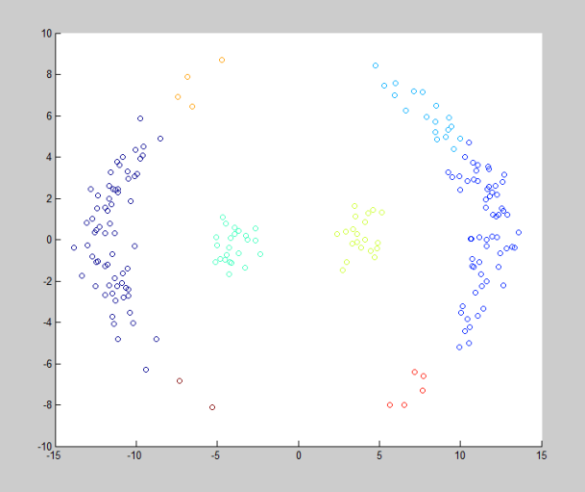
Trail A



Trail B



Trail C



In the mean-shift method. All the 3 trails have proper clustering, we can see the colors show that the mixing of the data is not very obvious.

However, in this method, the continuity of the cluster is not very good, which means several clusters should converge together because they stay very close to each other.

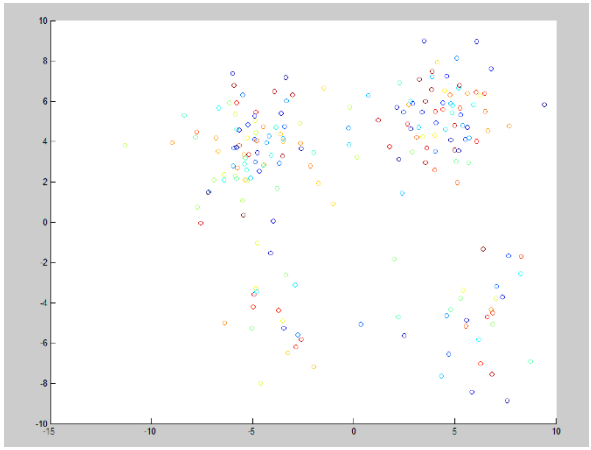
This might be the result of improper initial value of bandwidth.

Question (c) is about the sensitivity of the mean-shift method to the bandwidth of kernel.

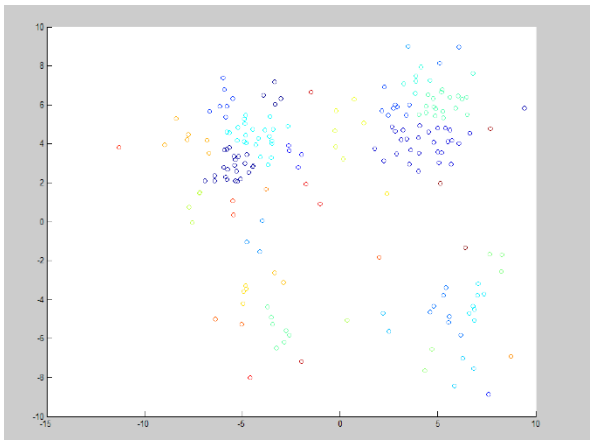
To test this result, I set different value of bandwidth h, including:

0.01, 0.3, 0.5, 1, 2, 5, 10,

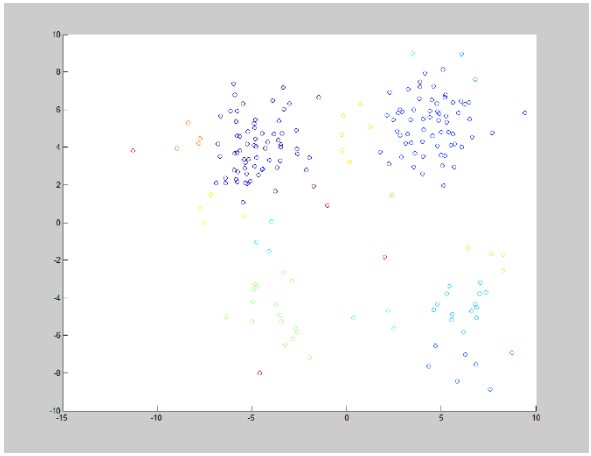
h = 0.01



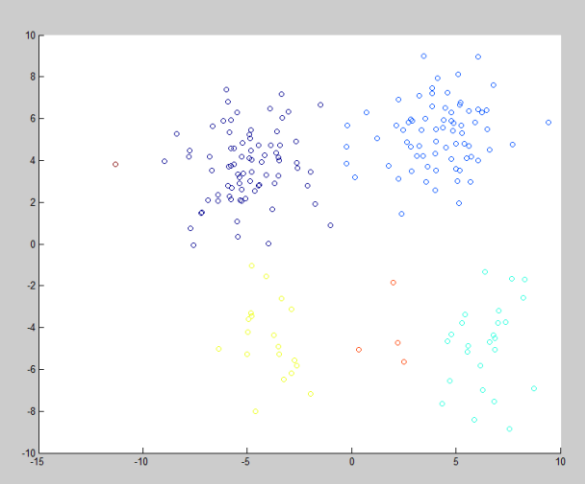
h = 0. 3



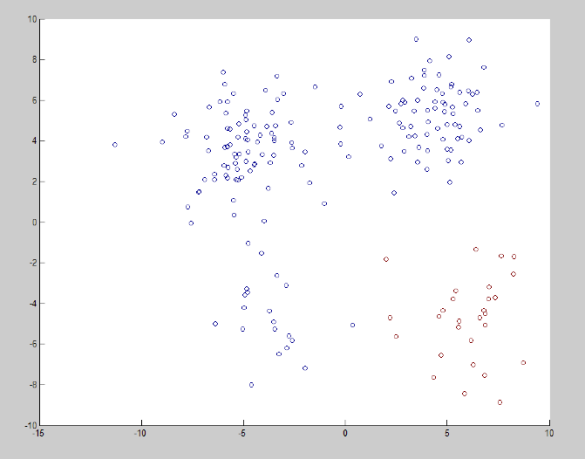
h = 0.5



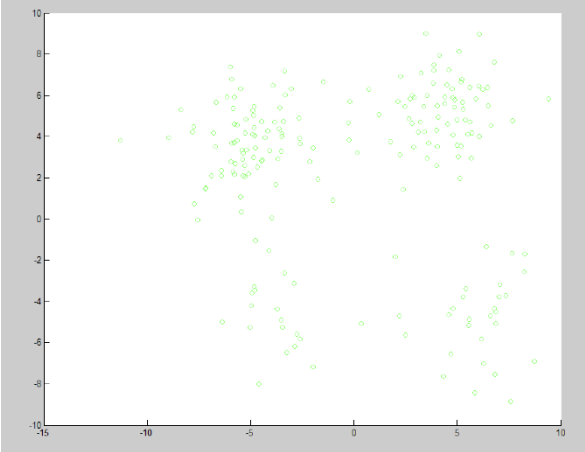
h = 1



h = 2



h = 5



From the results of different bandwidth, we can see that when bandwidth is not long enough, the points are clustered in very sparse scales. And the clusters are more than the correct results.

We can see the bandwidth around 1 is good enough to have a proper cluster result. That might be related to the range of the data points in the trail.

And when the h value is too much, the cluster number will decrease. For h = 2, there are only 2 clusters, and h = 5, only 1 cluster.

So this might be the sensitivity of the kernel bandwidth to the clustering results. As h increases, the cluster range increases, and the number of clusters may decrease. i.e. The cluster will converge as h increases, and separate when h decreases.