模糊控制导论

苏临之
sulinzhi029@nwu.edu.cn

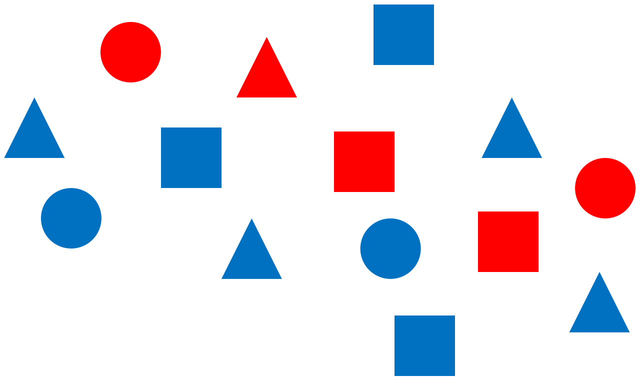
模糊控制导论纲要

- 模糊控制基本概念
- 模糊集合及其运算
- 模糊关系的数学表示和运算
- 模糊控制逻辑基础与推理运算
- **模糊C均值聚类法**
- 科技文献书写和阅读

聚类

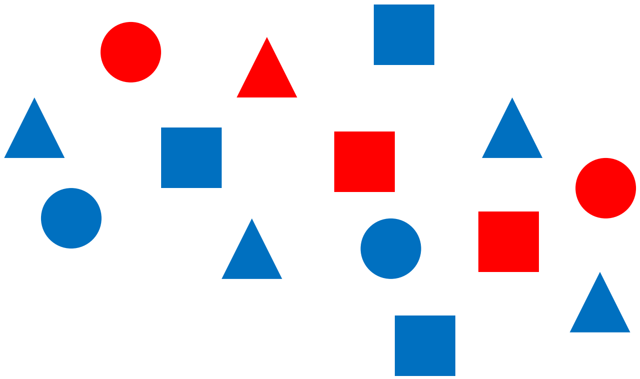
- 把性质相似的事物聚在一起形成一个类，这个过程称之为聚类（Clustering）。

- 请思考：右侧的这些图形可以聚成几类？都有哪几类？



聚类

- 把性质相似的事物聚在一起形成一个类，这个过程称之为聚类（Clustering）。
- 右侧的图形中，如果就形态性质聚类则可以聚成三角、圆和方三类；如果就颜色性质聚类，则可以聚成红色和蓝色两类。当然可以按照别的性质进行聚类。



数据性质的量化

- 计算智能中，所有事物的性质都是需要量化的。如颜色可以用通道灰度数值来表示，位置可以用坐标来表示。而形态则需要用更复杂的深度特征来表示。
- 需要注意的是，性质的数据化表示不一定是某一个数字，有可能是一个多维的向量，甚至是矩阵或其他多维阵列。例如常用的颜色性质数据就是由RGB构成的三维向量。这些被量化的性质可以通过计算来进行聚类。

聚类方法

- 常用的聚类方法有两种：一种是K均值法聚类(K-means，KM)，它是硬聚类法；另一种是模糊C均值聚类法(Fuzzy c-means，FCM)，它是软聚类法。其余算法基本上是对它们的改进处理。

基本聚类法

硬聚类：KM

软聚类：FCM

KM算法基本思想

- KM算法的基本思想是：把n个数据x_j（j=1,2…,n)分为m个类组H_i(i=1,2,…,m)，并求每组H_i的聚类中心c_i，使目标函数J达到最小。

- 在基本思想里，有两个比较重要的点值得关注：聚类中心c_i 和目标函数J。算法本身就是不断通过迭代的方式，通过优化代价函数J的寻找诸聚类中心c_i，进一步明确各个元素所属的类别。

聚类中心

- 聚类中心可以定义为当前分类时每一类的数据的算数平均值。假设类别H_i中元素的个数是$|H_i|$，则聚类中心c_i可以计算如下：

$$c_i = \frac{1}{|H_i|} \sum_{x \in H_i} x$$

目标函数

- 目标函数J是对当前分类结果的评价函数。如果函数值偏大，则分类结果较差；如果函数值较小，则分类结果较好。根据这个要求，可以用各个元素到聚类中心的欧几里得距离总和来构建J。

$$J=\sum_{i=1}^{m}\sum_{x\in H_i}\left\|x-c_i\right\|^2$$

聚类中心和目标函数的初始化

- KM是迭代算法，因此首先必须要对聚类中心进行初始化。由于聚类的类别数m已知，因此常用的方法是从中任意选择m个元素作为聚类中心。然后通过公式计算初始目标函数值。这里使用上标(1)表示第1次迭代过程。

$$c_i^{(1)} = t_i \quad t_i = \mathrm{rand}(x)$$

$$J^{(1)} = \sum_{i=1}^{m} \sum_{x \in H_i} \left\| x - c_i^{(1)} \right\|^2$$

元素类别的划分

- 当进行第k次迭代计算时，得到了诸聚类中心$c_i^{(k)}$，这样就可以计算每个元素x到诸聚类中心的距离$d_i^{(k)}$。x到哪一类中心的距离最短，则就把x划归到哪一类。

$$d_i^{(k)} = \left\| x - c_i^{(k)} \right\|$$

$$d_r^{(k)} = \min\left\{ d_i^{(k)} \right\}$$

$$x \in H_r^{(k)}$$

聚类中心的更新

- 对所有元素x进行分类以后，每一个元素都有类别标签，此时需要根据这个标签更新聚类中心。并计算新的目标函数值。

$$c_i^{(k+1)} = \frac{1}{\left|H_i^{(k)}\right|} \sum_{x \in H_i^{(k)}} x$$

$$J^{(k+1)} = \sum_{i=1}^{m} \sum_{x \in H_i^{(k)}} \left\| x - c_i^{(k+1)} \right\|^2$$

迭代终止条件

- 上述过程反复循环迭代。为了结束算法循环，需要一个终止条件。最常用的终止条件就是目标函数值几乎不再变化，这里使用足够小的正数δ来体现。

$$\left| J^{(k+1)} - J^{(k)} \right| \le \delta$$

- 如果上述条件满足，则退出迭代循环，输出每个元素的标签作为最终聚类结果；如果上述条件不满足，则令$k:=k+1$，继续循环计算距离并更新聚类中心和目标函数，直到终止条件满足为止。

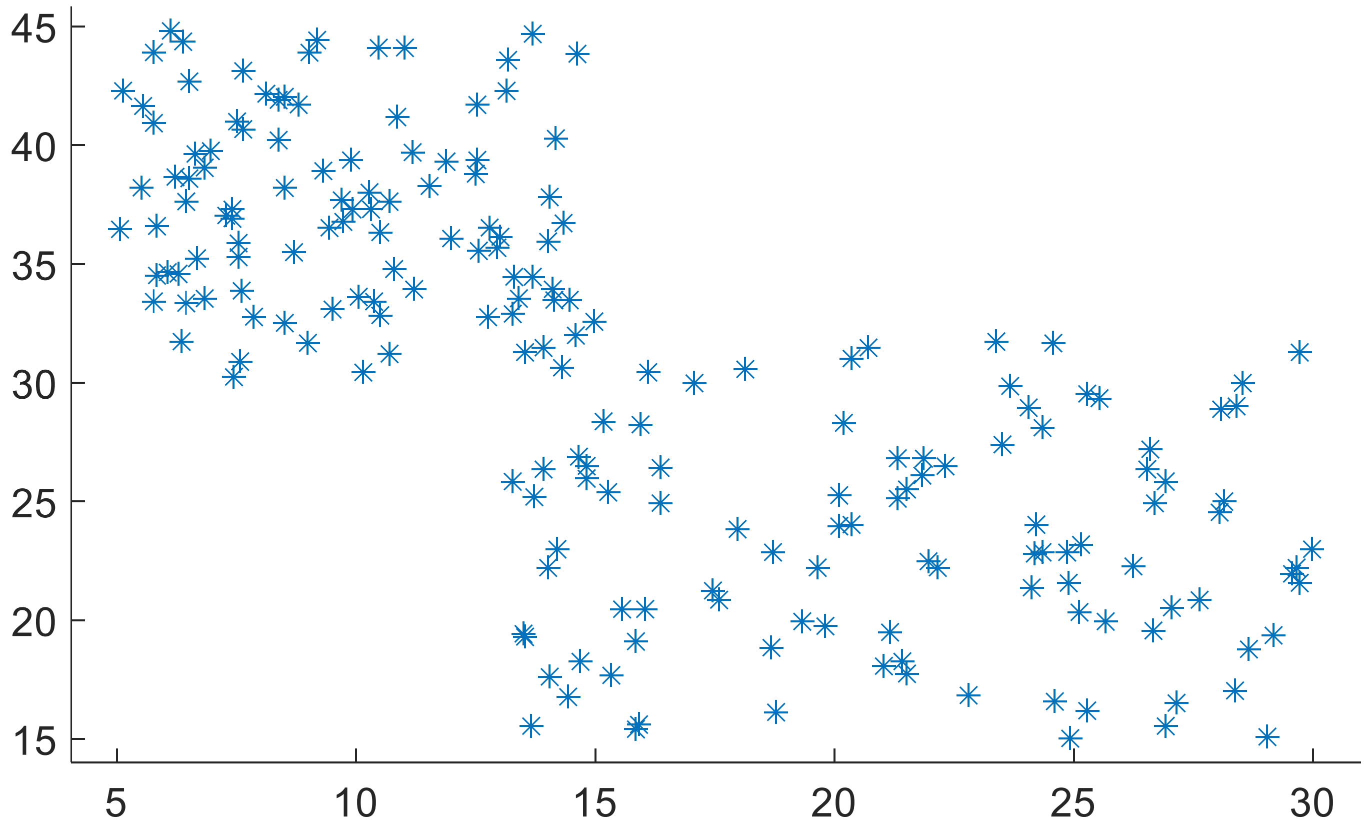
KM子程序的调用

- MATLAB自带kmeans函数，可以直接调用。有以下几种常用调用方式。

```
U=kmeans(x,m)

[U,c]=kmeans(x,m)

[U,c,sum_dist]=kmeans(x,m)

[U,c,sum_dist,dist]=kmeans(x,m)
```

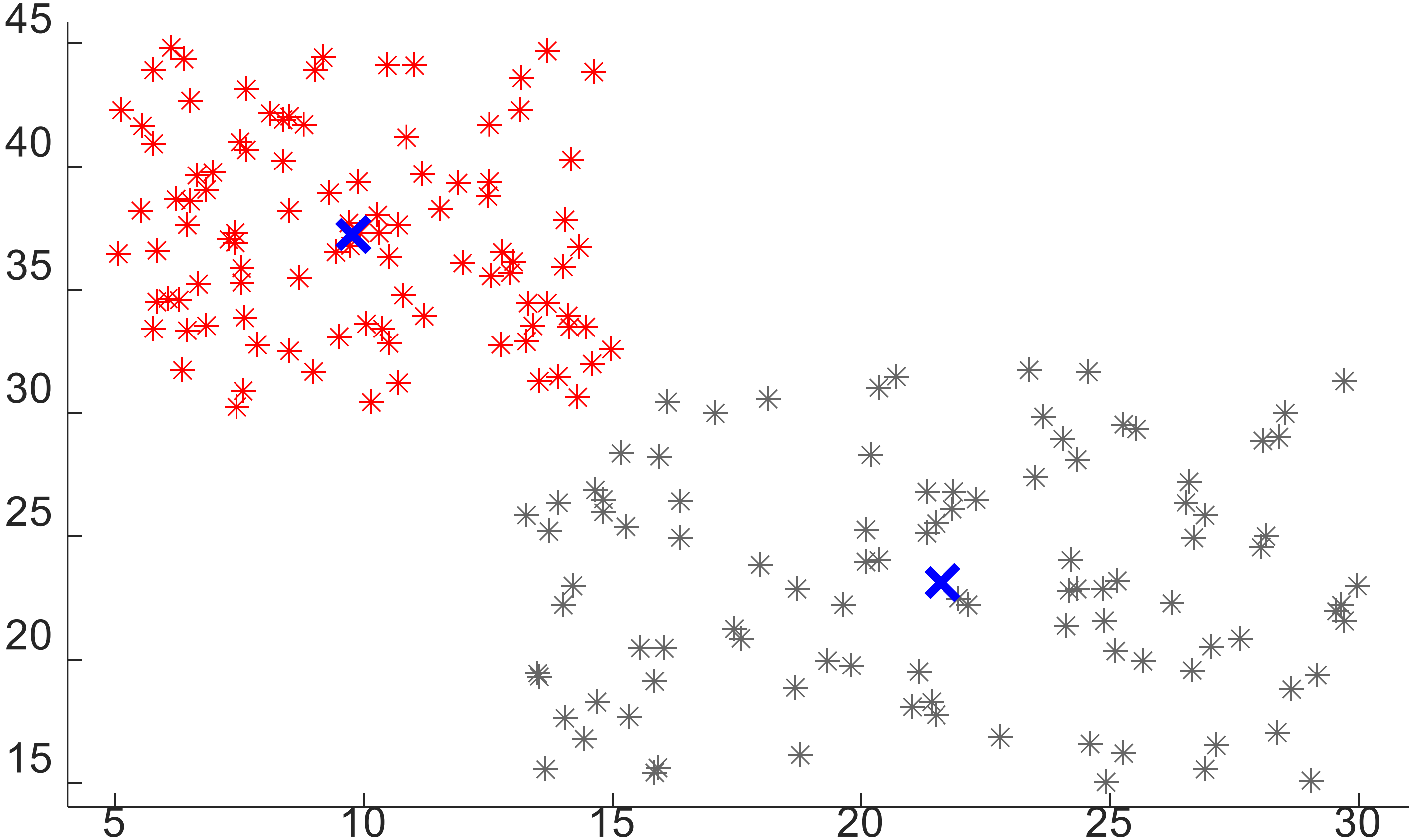
实验数据1

- 第一组数据，区分大，数量较少，以平面点位置为特征：



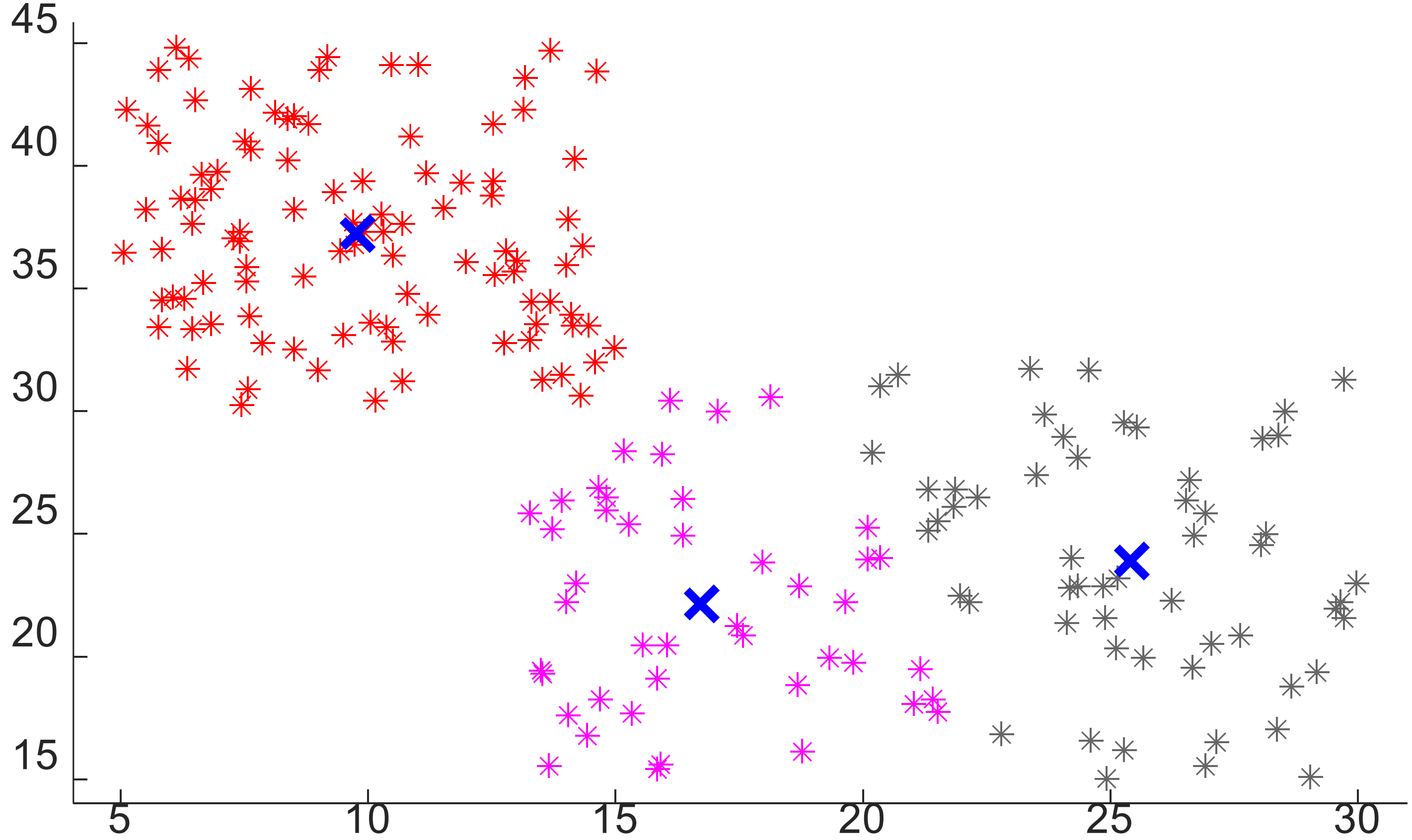
实验结果

- 聚类数m=2结果如下：



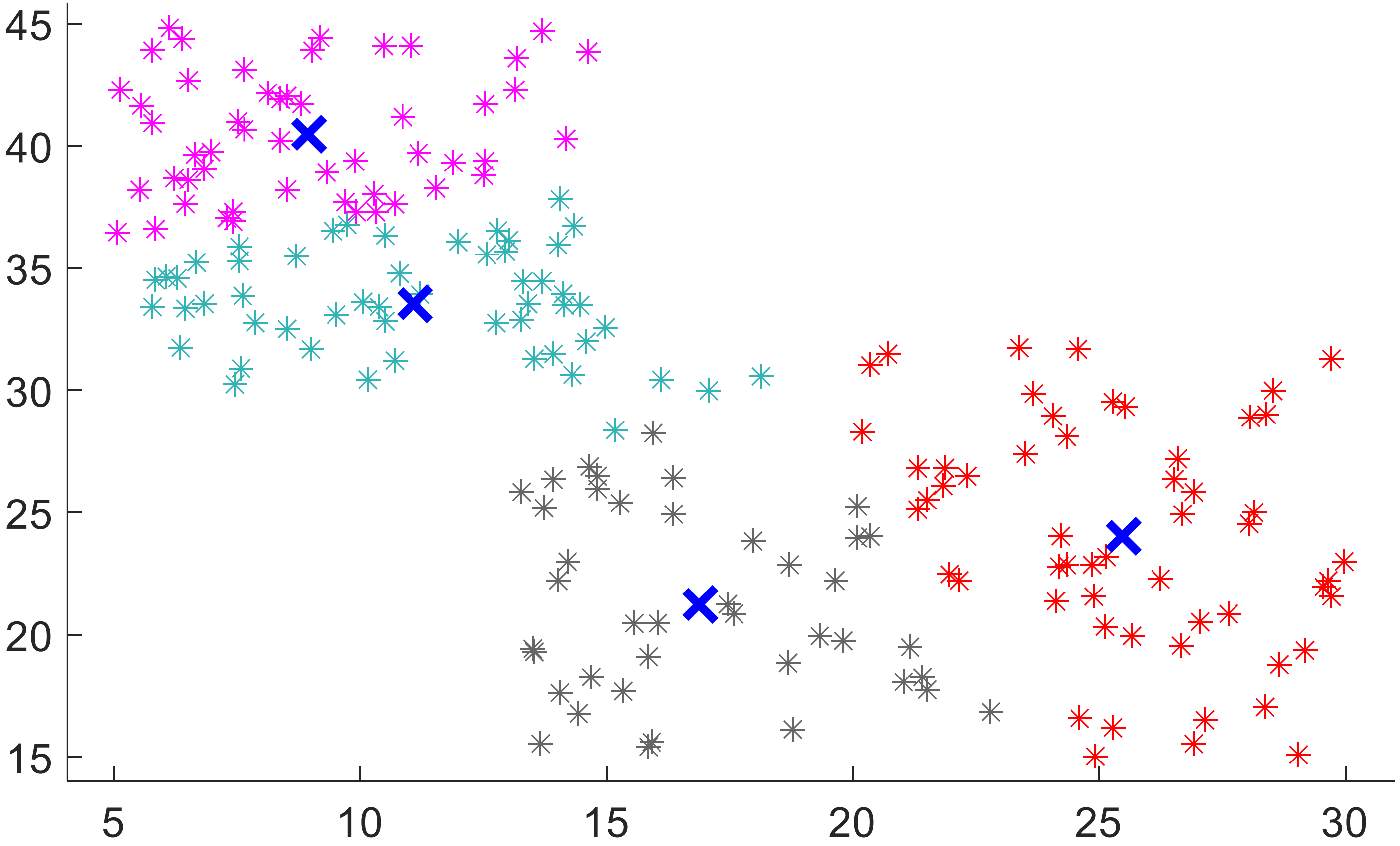
实验结果

- 聚类数m=3结果如下：



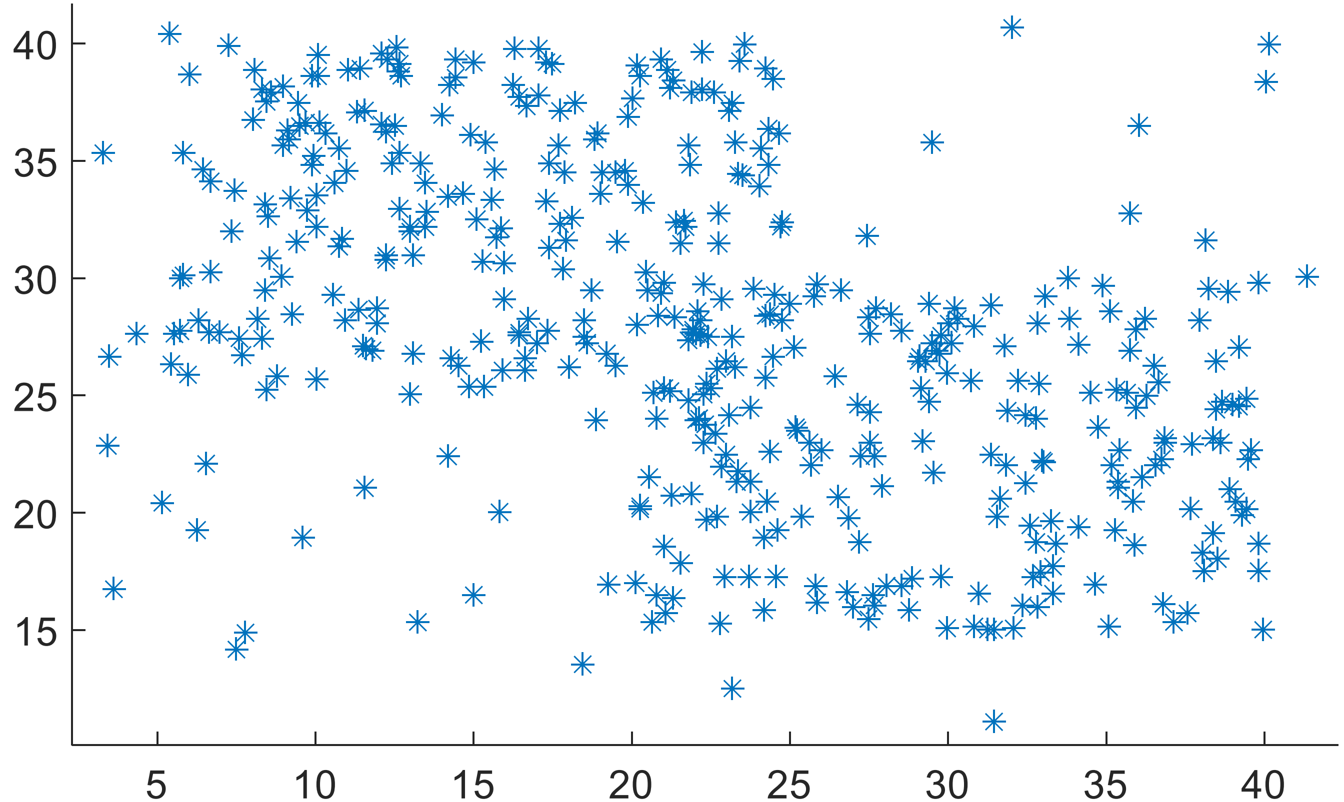
实验结果

- 聚类数m=4结果如下：



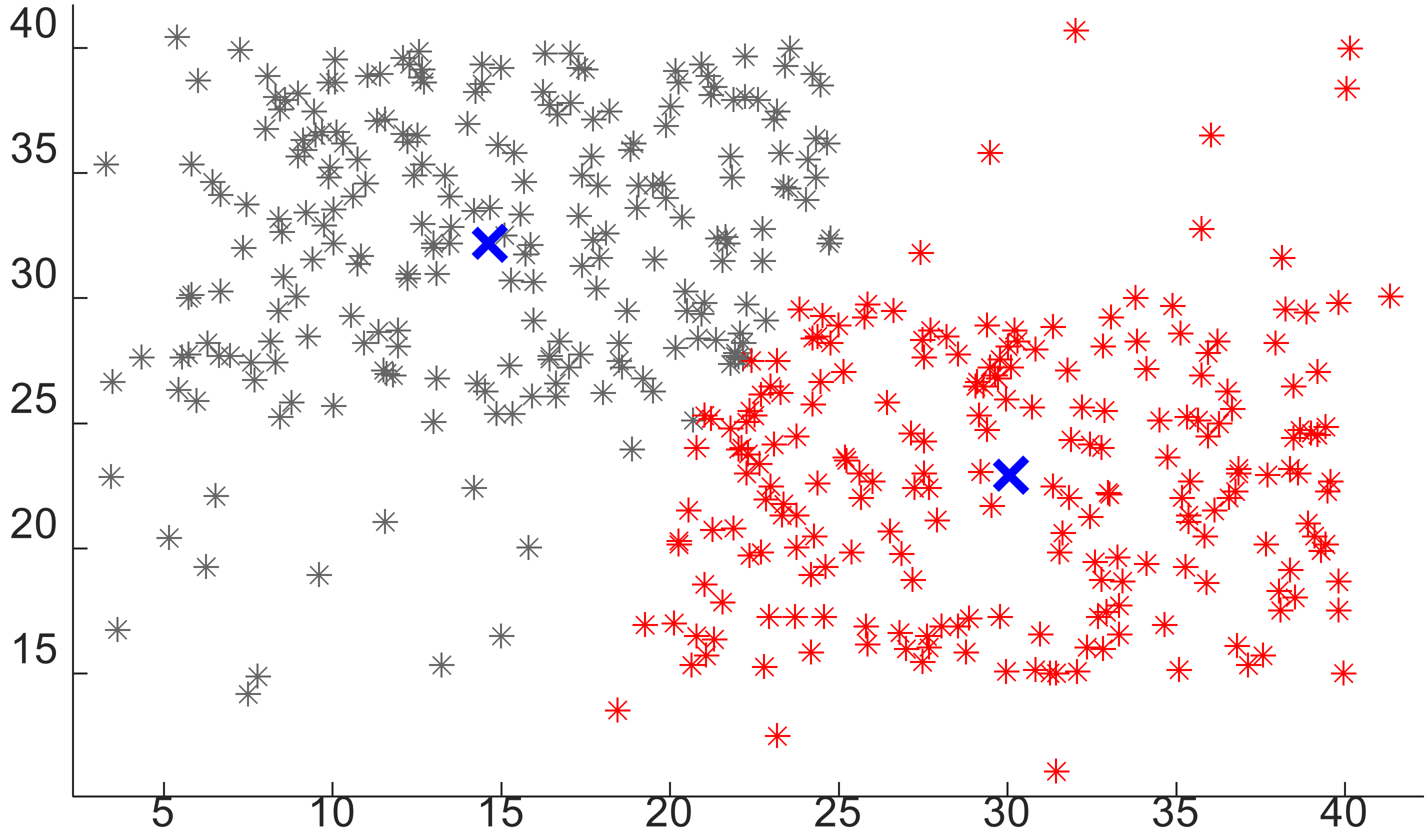
实验数据2

- 第二组数据，区分度略小，数量略多：



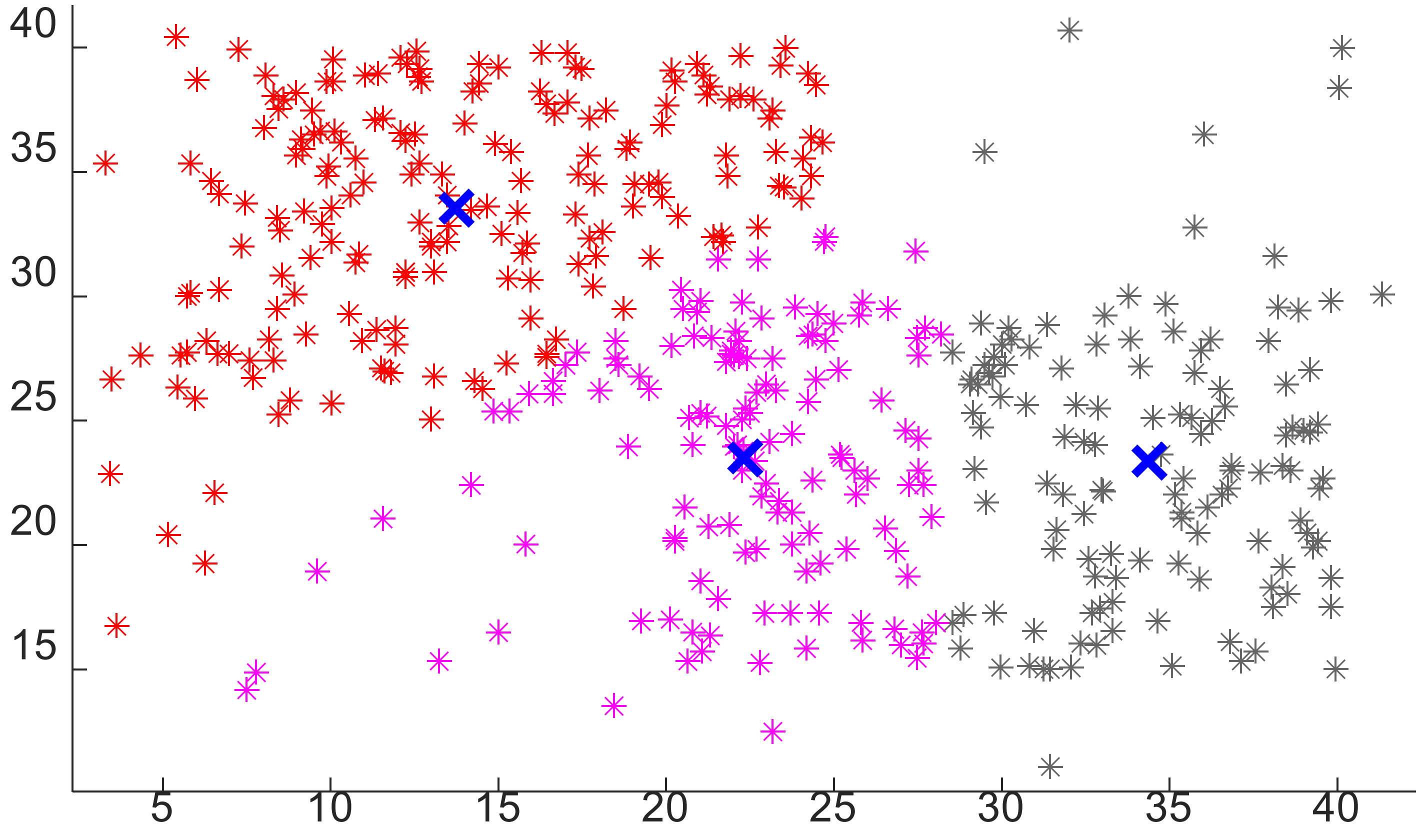
实验结果

- 聚类数m=2结果如下：



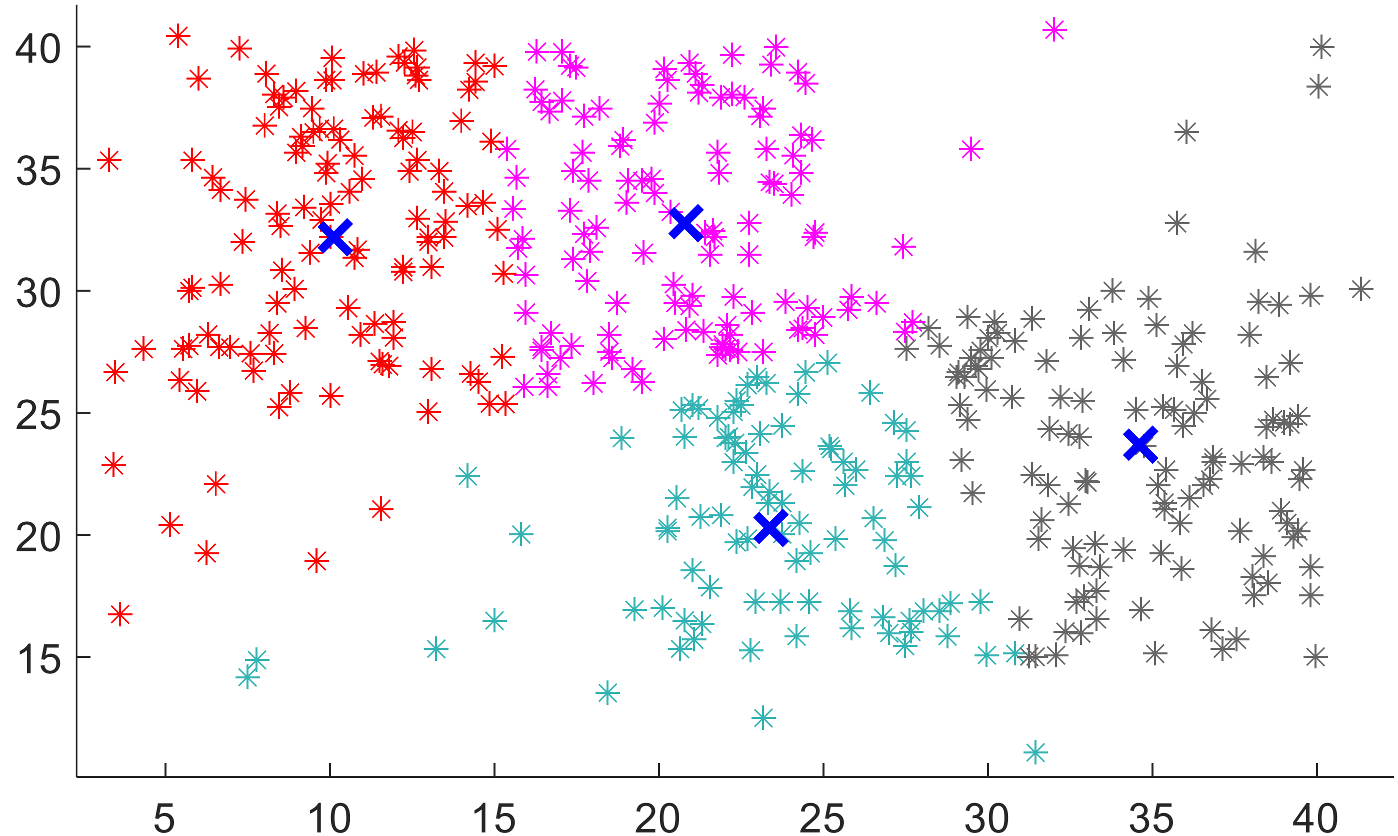
实验结果

- 聚类数*m*=3结果如下：



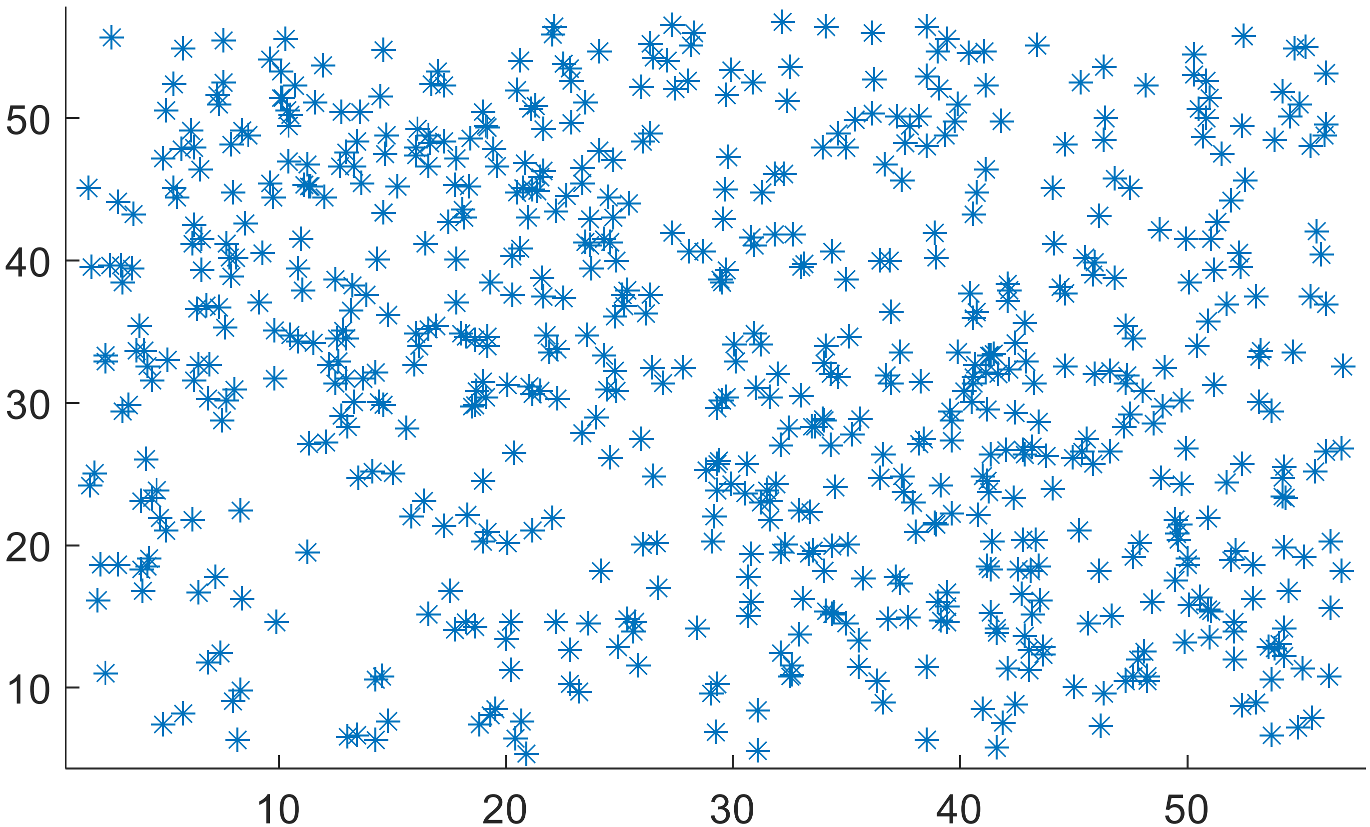
实验结果

- 聚类数m=4结果如下：



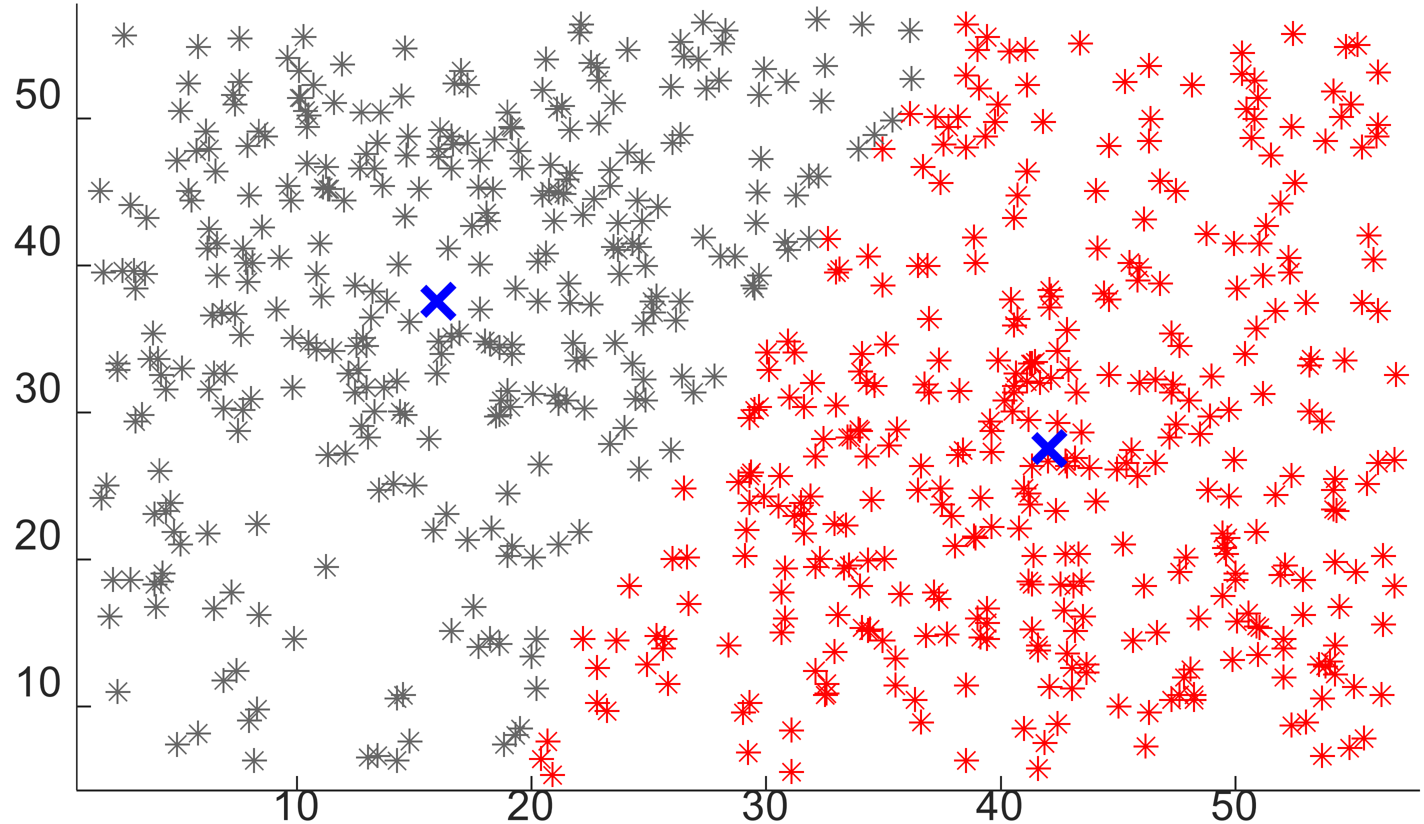
实验数据3

- 使用另一组区分度更小、点数更多的数据如下：



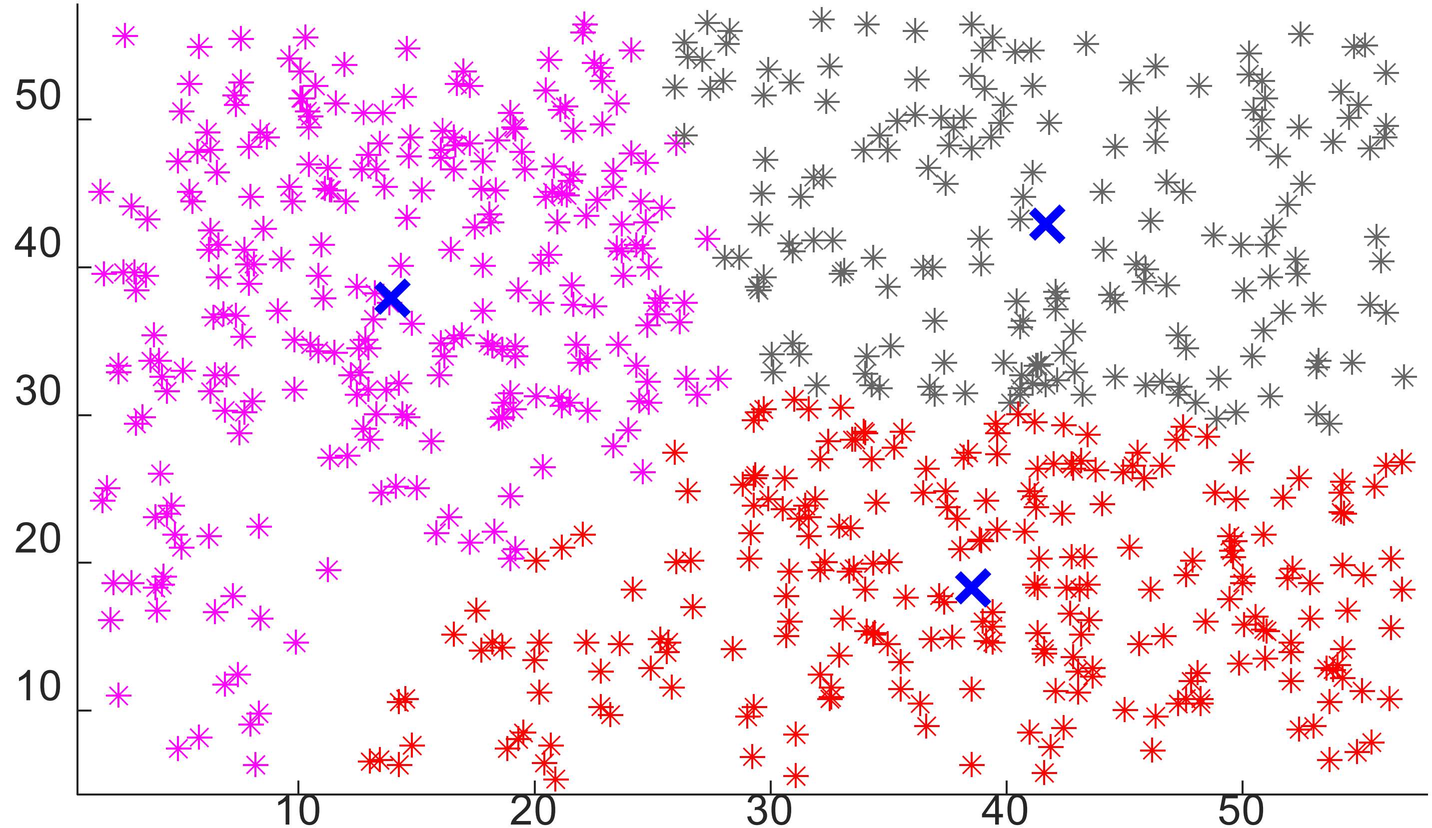
实验结果

- 聚类数*m*=2结果如下：



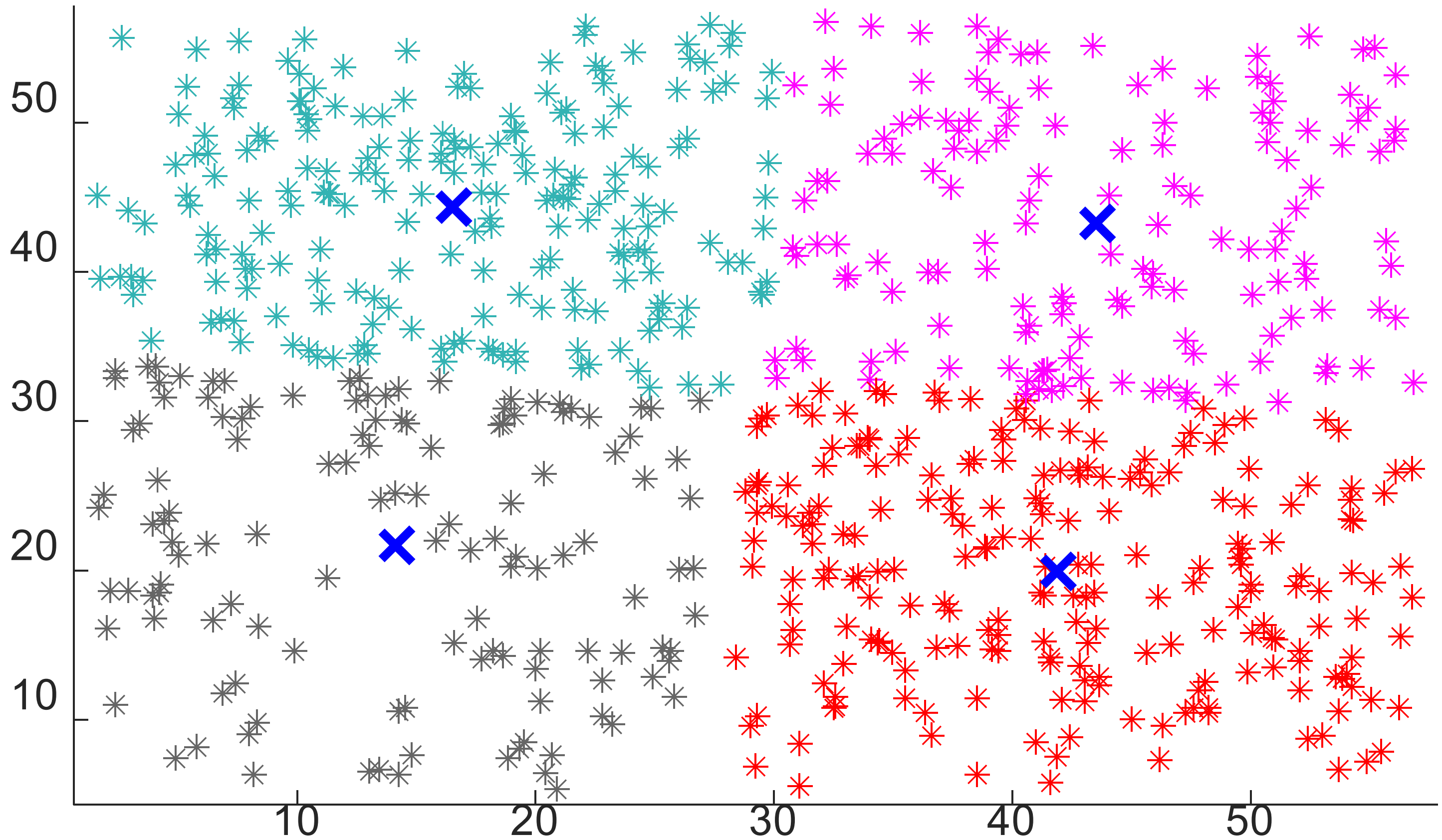
实验结果

- 聚类数*m*=3结果如下：



实验结果

- 聚类数m=4结果如下：



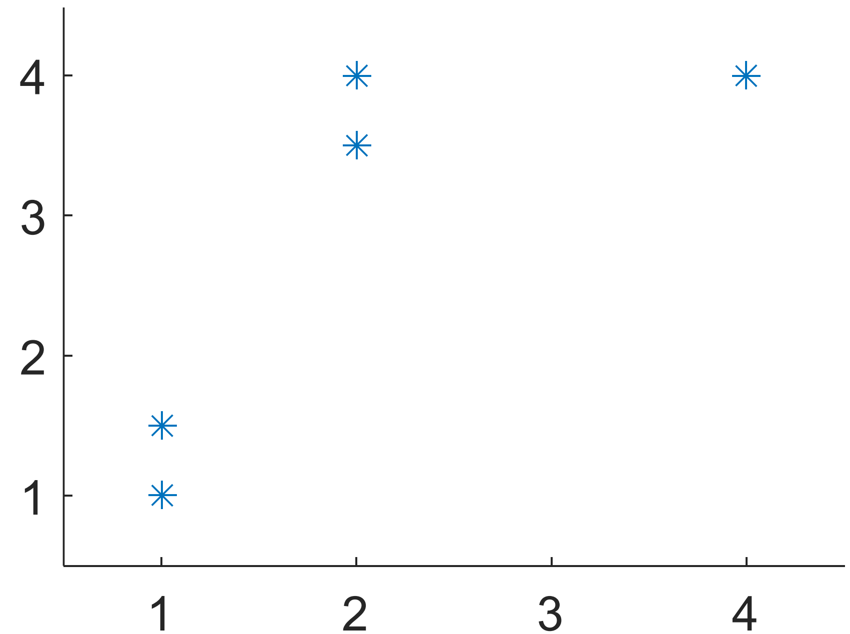
KM算法中的隶属度矩阵

- 最后对元素的聚类标签，实际上相当于生成一个隶属度矩阵，它是一个布尔矩阵。例如有5个元素，分3类，则它的隶属度矩阵***U***如下：

$$x_1 = (1,1) \quad x_2 = (1,1.5) \quad x_3 = (2,4)$$

$$x_4 = (2,3.5) \quad x_5 = (4,4)$$

$$\boldsymbol{U} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$



目标函数的另一种形式

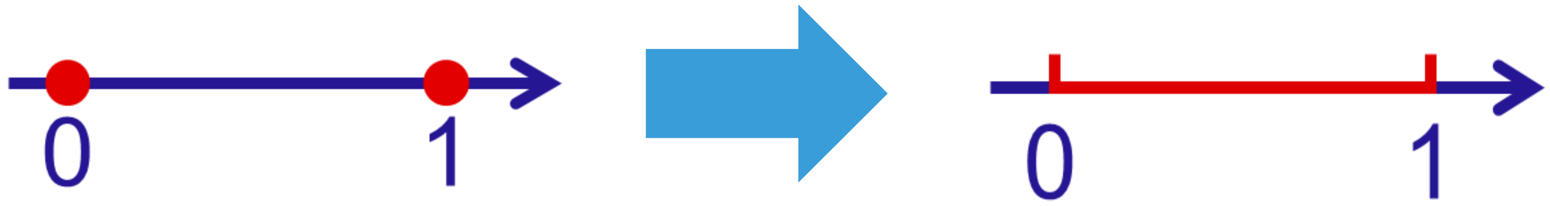
- 利用隶属度矩阵，目标函数J可写成另一种形式。设元素j对于第i类的隶属度值是u_{ij}，则有：

$$J = \sum_{i=1}^{m} \sum_{j=1}^{n} u_{ij} \left\| x_j - c_i \right\|^2$$

$$\sum_{i=1}^{m} u_{ij} = 1$$

从KM算法到FCM算法

- 如果隶属度矩阵取值范围不是{0,1}而是[0,1]，就得到了FCM算法。因此KM被称为硬聚类法，而FCM则是软聚类法。



FCM算法的目标函数

- FCM算法的目标函数如下，其隶属度取值范围是[0,1]。$\alpha\in(1,+\infty)$是一个加权参数。因此本质上就是在隶属度和为1的条件下求J的最小值。

$$J=\sum_{i=1}^{m}\sum_{j=1}^{n}u_{ij}^{\alpha}\left\|x_j-c_i\right\|^2$$

$$\sum_{i=1}^{m}u_{ij}-1=0\quad \forall j=1,2,\ldots,n$$

拉格朗日目标函数构造

- 使用拉格朗日乘子法构建新的目标函数。这个函数中包括三类变量，u_{ij}、c_i和λ_j。为了求出这个函数的最小值，需要对其中的每一个变量求偏导数，并使得偏导数等于0。

$$\bar{J} = \sum_{i=1}^{m}\sum_{j=1}^{n} u_{ij}^{\alpha}\left\|x_j - c_i\right\|^2 + \sum_{j=1}^{n}\lambda_j\left(\sum_{i=1}^{m}u_{ij} - 1\right)$$

聚类中心的更新公式

- 对聚类中心求偏导数，并令其为0，有：

$$\frac{\partial \bar{J}}{\partial c_i} = 2\sum_{j=1}^{n} u_{ij}^{\alpha}\left(c_i - x_j\right) = 0$$

$$\Rightarrow c_i = \frac{\sum_{j=1}^{n} u_{ij}^{\alpha} x_j}{\sum_{j=1}^{n} u_{ij}^{\alpha}}$$

隶属度的更新公式

- 对隶属度求偏导数，并令其为0，可以得到隶属度更新原始公式。这个式子里含有参数λ_j，必须进一步消掉。

$$\frac{\partial \bar{J}}{\partial u_{ij}} = \alpha \left\| x_j - c_i \right\|^2 u_{ij}^{\alpha-1} + \lambda_j = 0$$

$$\Rightarrow u_{ij} = \left(\frac{-\lambda_j}{\alpha \left\| x_j - c_i \right\|^2} \right)^{\frac{1}{\alpha-1}}$$

隶属度的更新公式

- 把这个表达式代入约束条件中，有：

$$\because \sum_{i=1}^{m} u_{ij} = 1 \quad u_{ij} = \left(\frac{-\lambda_j}{\alpha \left\| x_j - c_i \right\|^2} \right)^{\frac{1}{\alpha-1}}$$

$$\therefore \sum_{i=1}^{m} \left(\frac{-\lambda_j}{\alpha \left\| x_j - c_i \right\|^2} \right)^{\frac{1}{\alpha-1}} = \left(\frac{-\lambda_j}{\alpha} \right)^{\frac{1}{\alpha-1}} \sum_{i=1}^{m} \left(\frac{1}{\left\| x_j - c_i \right\|^{\frac{2}{\alpha-1}}} \right) = 1$$

隶属度的更新公式

$$\left(\frac{-\lambda_j}{\alpha}\right)^{\frac{1}{\alpha-1}} \sum_{i=1}^{m} \left(\frac{1}{\left\| x_j - c_i \right\|^{\frac{2}{\alpha-1}}} \right) = 1$$

$$\Rightarrow \left(\frac{-\lambda_j}{\alpha}\right)^{\frac{1}{\alpha-1}} = \frac{1}{\sum_{i=1}^{m} \left(\frac{1}{\left\| x_j - c_i \right\|^{\frac{2}{\alpha-1}}} \right)} = \frac{1}{\sum_{k=1}^{m} \left(\frac{1}{\left\| x_j - c_k \right\|^{\frac{2}{\alpha-1}}} \right)}$$

隶属度的更新公式

- 代回更新公式，避免变量冲突，将求和变量改为k。

$$u_{ij}=\left(\frac{-\lambda_j}{\alpha\left\|x_j-c_i\right\|^2}\right)^{\frac{1}{\alpha-1}}=\left(\frac{-\lambda_j}{\alpha}\right)^{\frac{1}{\alpha-1}}\cdot\frac{1}{\left\|x_j-c_i\right\|^{\frac{2}{\alpha-1}}}$$

$$=\frac{1}{\sum_{k=1}^{m}\left(\frac{1}{\left\|x_j-c_k\right\|^{\frac{2}{\alpha-1}}}\right)}\cdot\frac{1}{\left\|x_j-c_i\right\|^{\frac{2}{\alpha-1}}}$$

隶属度的更新公式

- 这样就得到了隶属度更新公式最终表达式：

$$u_{ij} = \frac{1}{\sum_{k=1}^{m} \left(\frac{\left\| x_j - c_i \right\|}{\left\| x_j - c_k \right\|} \right)^{\frac{2}{\alpha - 1}}}$$

更新公式总结

- 在FCM迭代循环中，聚类中心和隶属度两者用以下公式更新。在实际操作中，可以通过随机数生成一批初始隶属度，然后不断更新。

$$c_i = \frac{\sum_{j=1}^{n} u_{ij}^{\alpha} x_j}{\sum_{j=1}^{n} u_{ij}^{\alpha}} \qquad u_{ij} = \frac{1}{\sum_{k=1}^{m} \left(\frac{\left\| x_j - c_i \right\|}{\left\| x_j - c_k \right\|} \right)^{\frac{2}{\alpha-1}}}$$

FCM计算举例

例1：设有4个数据，利用FCM算法聚成2类（加权指数参数是2）。已知某次迭代中隶属度如下，求下一次迭代聚类中心c_1'和c_2'。

$$x_1=(2,2)\quad x_2=(2,3)\quad x_3=(4,8)\quad x_4=(4,7)$$

$$u_{11}=0.1\quad u_{12}=0.3\quad u_{13}=0.5\quad u_{14}=0.7$$

$$u_{21}=0.9\quad u_{22}=0.7\quad u_{23}=0.5\quad u_{24}=0.3$$

FCM计算举例

$$c_1' = \frac{0.1^2\times(2,2)+0.3^2\times(2,3)+0.5^2\times(4,8)+0.7^2\times(4,7)}{0.1^2+0.3^2+0.5^2+0.7^2}$$

$$=(3.76,6.81)$$

$$c_2' = \frac{0.9^2\times(2,2)+0.7^2\times(2,3)+0.5^2\times(4,8)+0.3^2\times(4,7)}{0.9^2+0.7^2+0.5^2+0.3^2}$$

$$=(2.41,3.49)$$

FCM的MATLAB实现

- 已经编写好的FCM子程序引用格式如下：

```
[center,U,obj_fcn] = FCM(x,m,options);
[center,U,obj_fcn] = FCM(x,m);
```

- 其中options是一个4×1列向量，诸分量含义如下：

options(1)：隶属度指数α>1

options(2)：最大迭代次数

options(3)：隶属度最小变化量，迭代终止条件

options(4)：每次迭代是否输出信息标志

- 当options缺省时，默认值为：[2,100,10^(-5),1]。

FCM程序应用

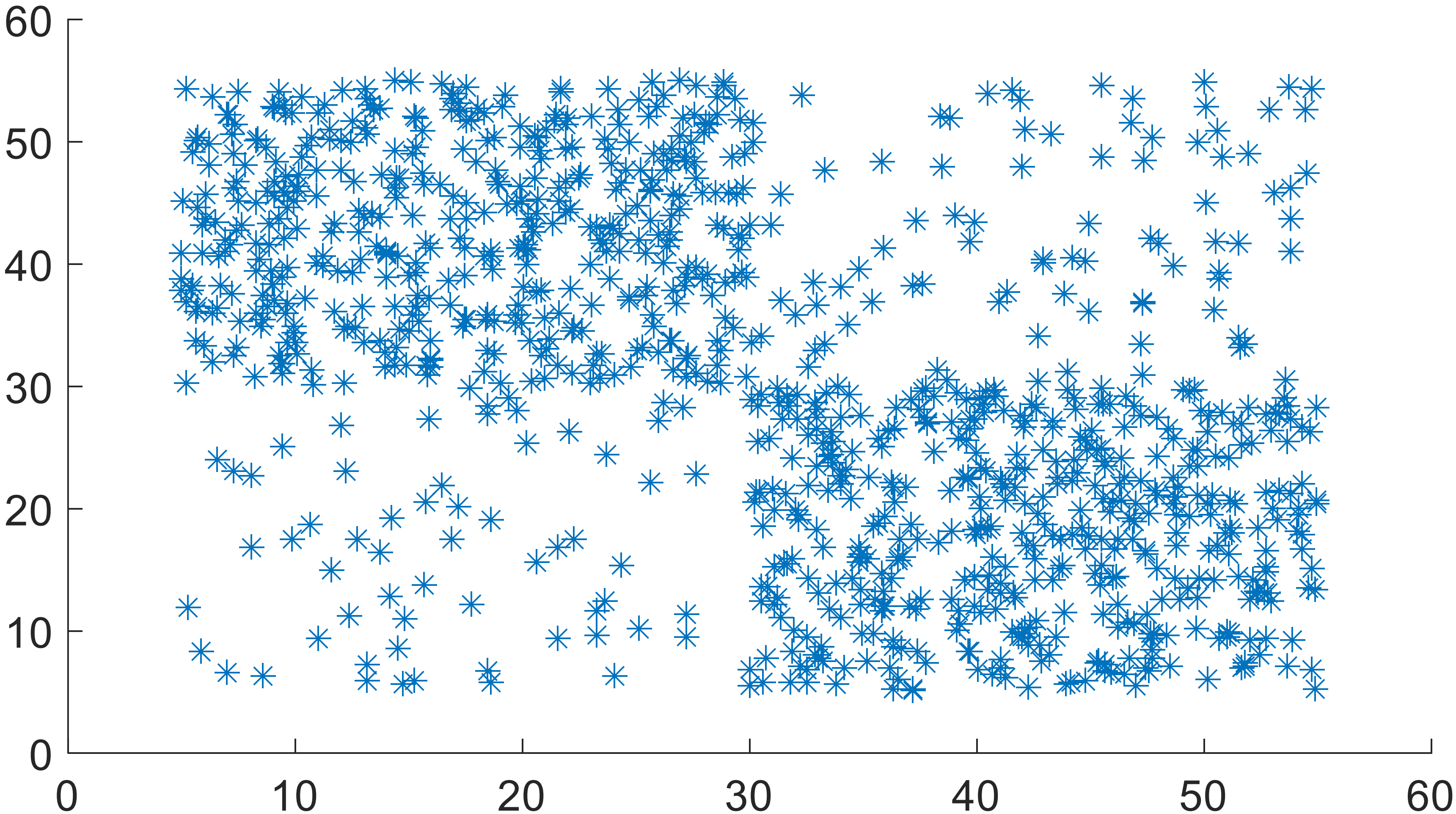
例2：在MATLAB环境下，使用随机数生成1000个疏密不一的二维数据，再利用FCM子程序进行聚类。要求聚成2、3、4和5类。将结果去掉外框，用不同颜色以散点形式显示出来。同时画出目标函数值随着迭代次数的变化图象。

随机数据的生成和显示

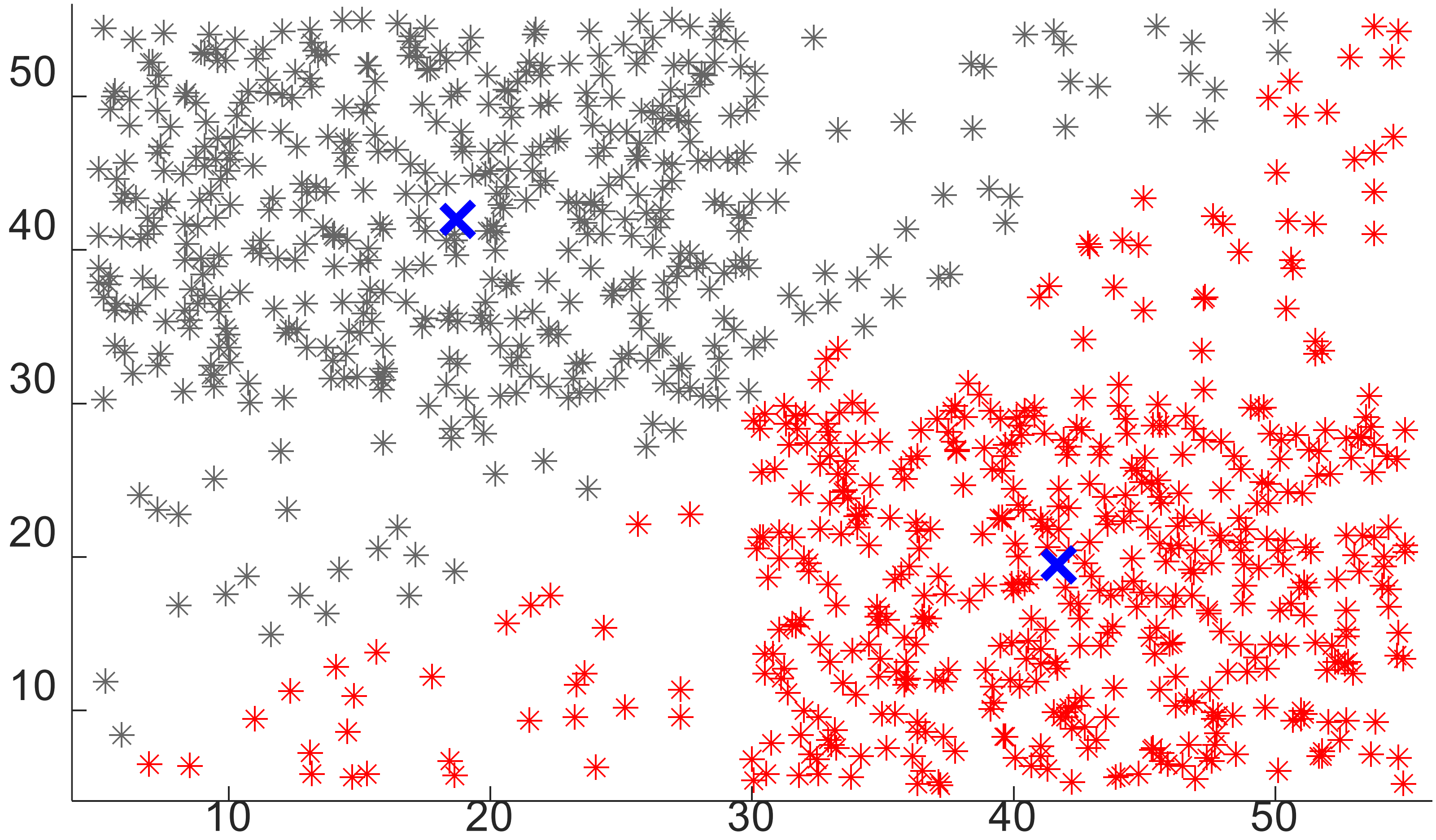
- 平面随机点的生成，需要生成一定范围内的随机横坐标和纵坐标。这里可以使用rand或者其类似函数。直接使用plot函数并附加对应的符号就可以描绘批量散点而不划线。疏密不一致的散点，只需进行多次叠加即可。

```
a1=5;a2=55;b1=5;b2=55;
x=a1+(a2-a1)*rand(100,1);
y=b1+(b2-b1)*rand(100,1);
figure;plot(x,y,'*');
box off;
```

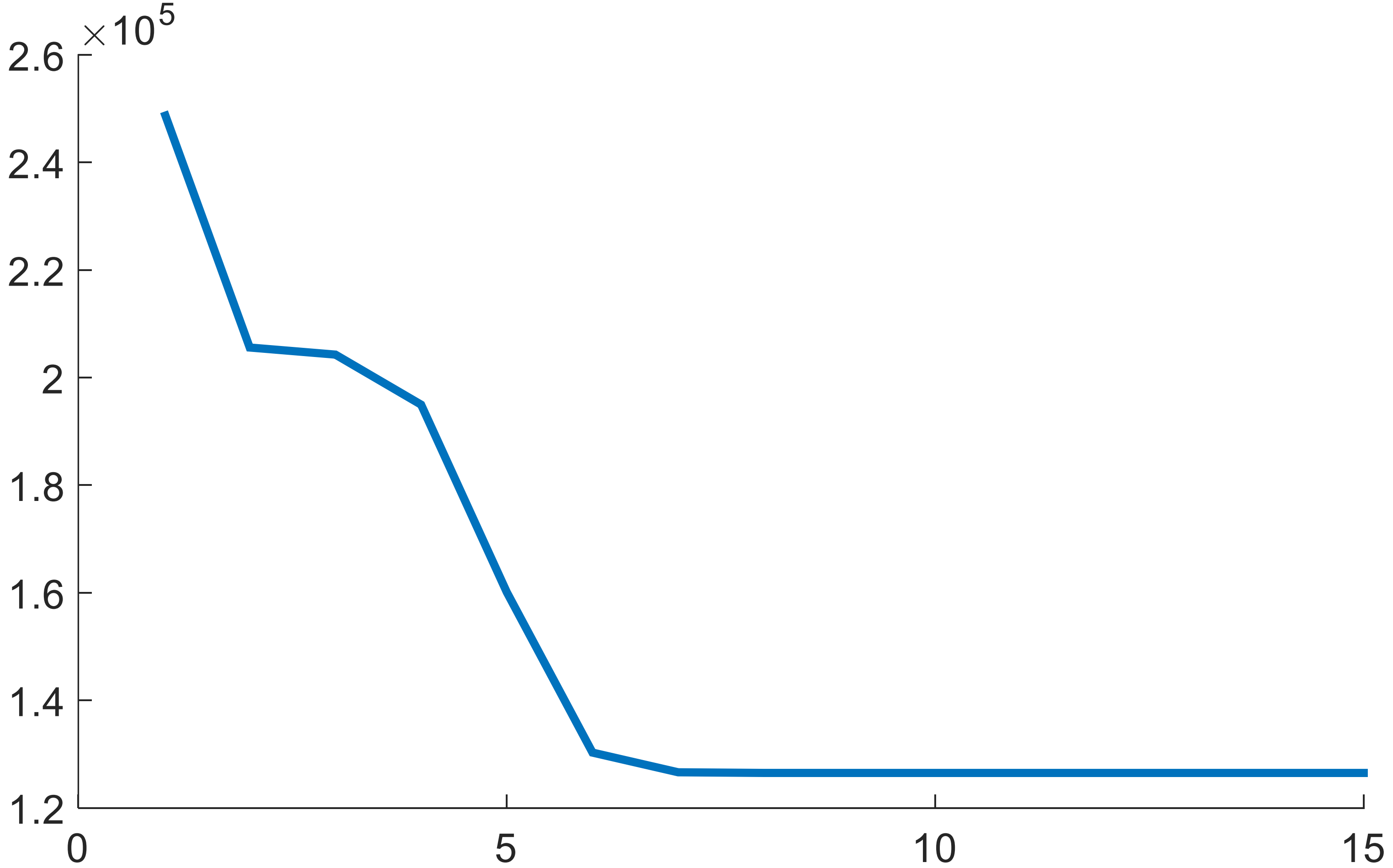
随机数据的生成和显示



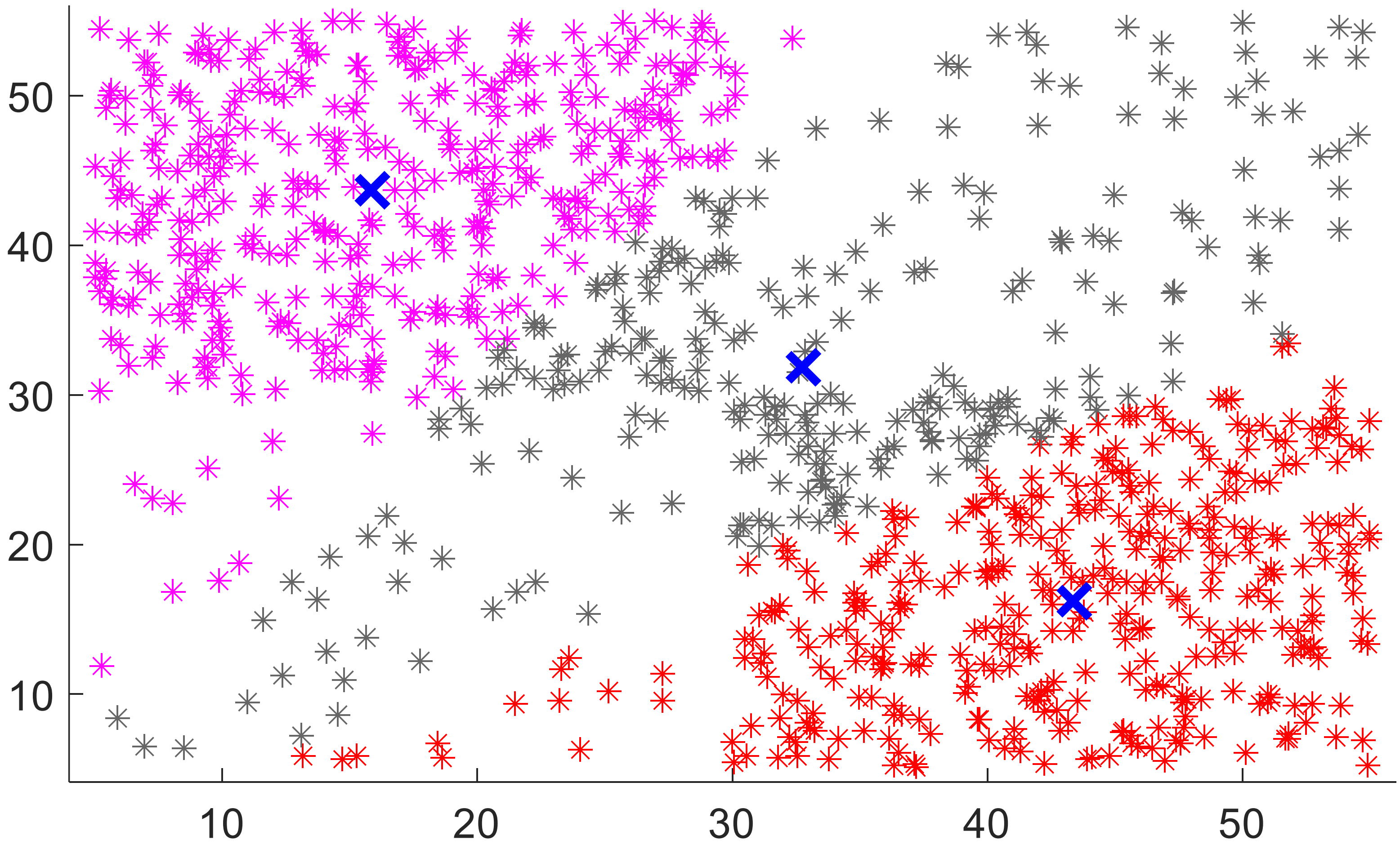
分2类结果



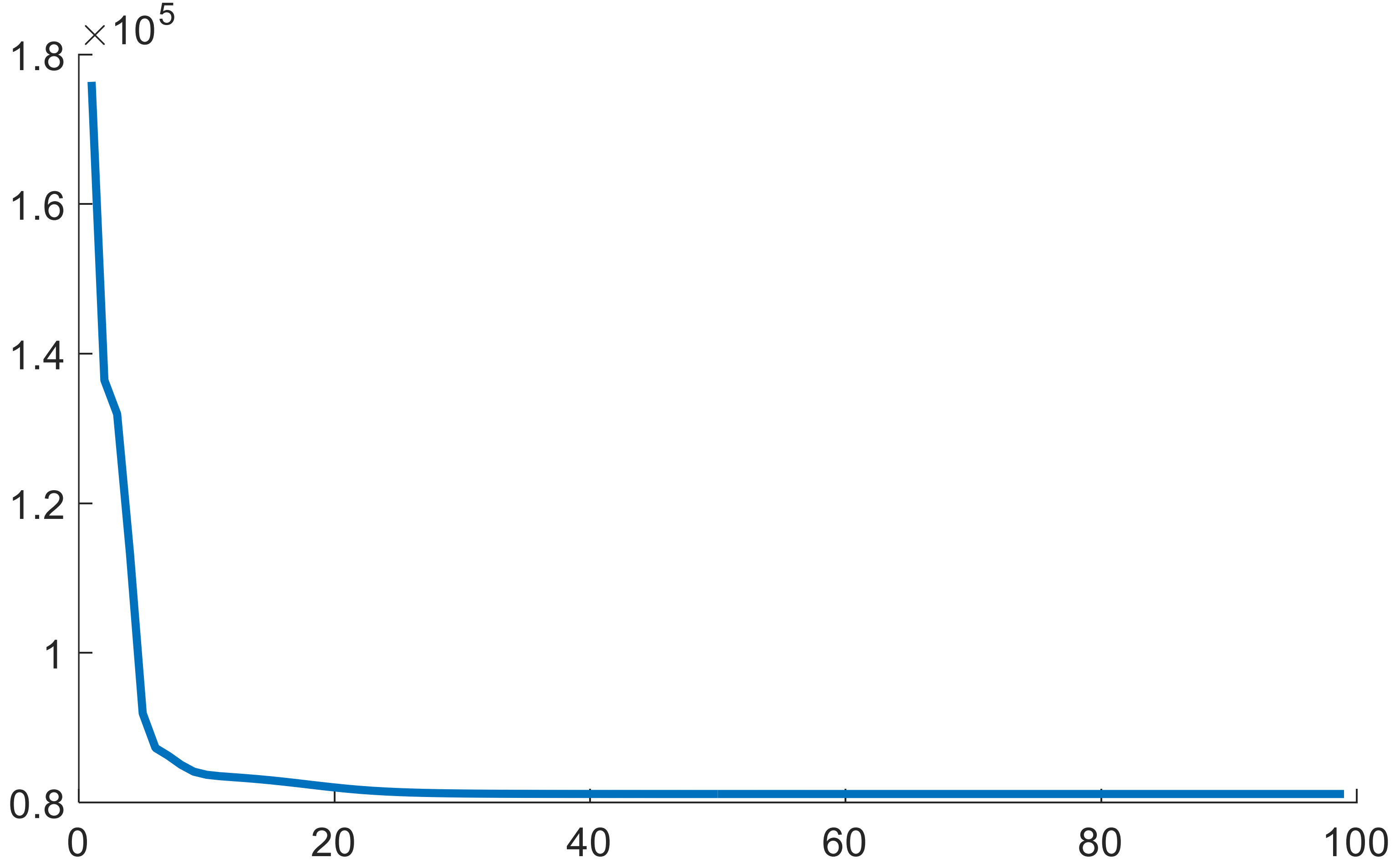
分2类结果



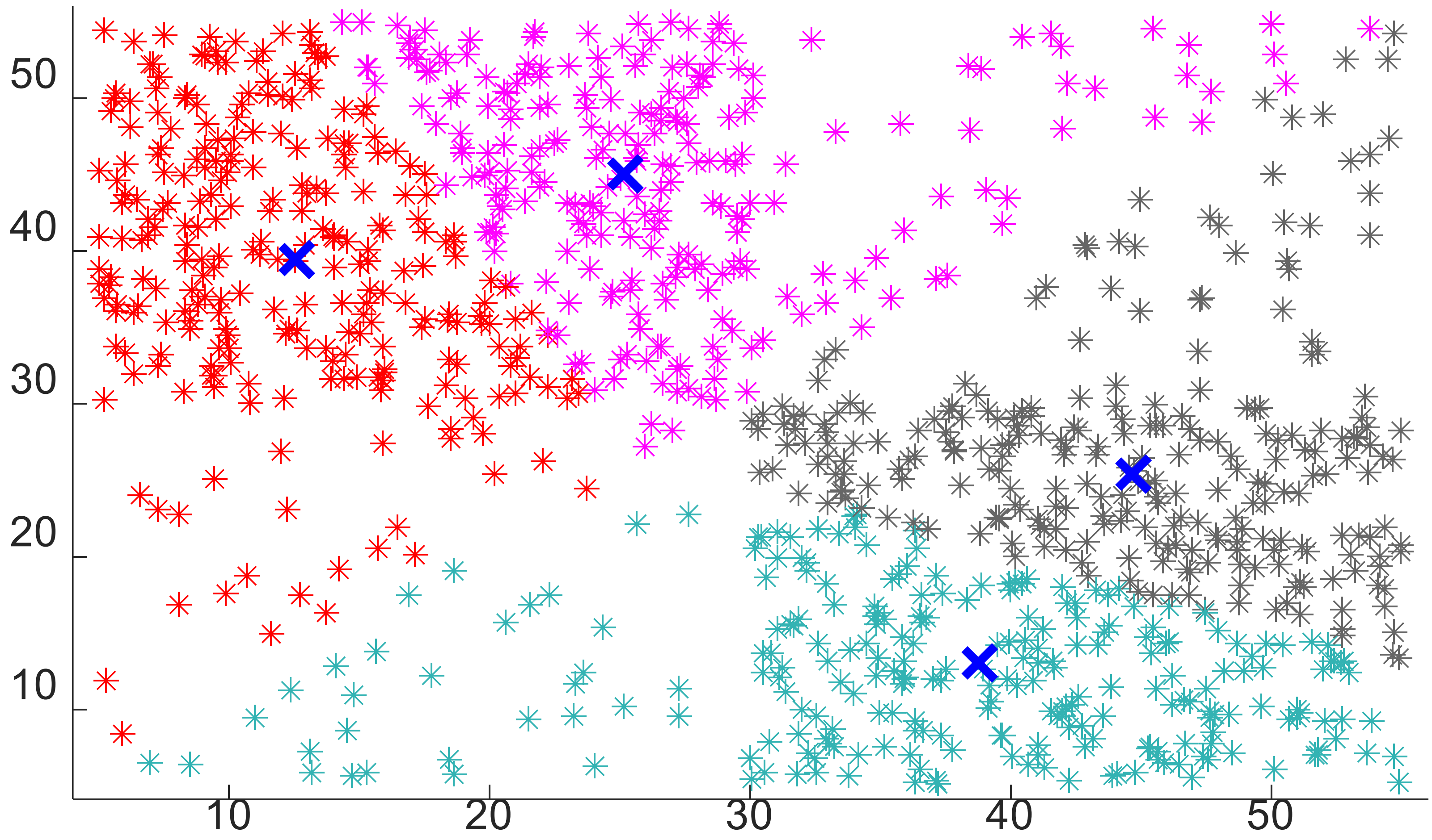
分3类结果



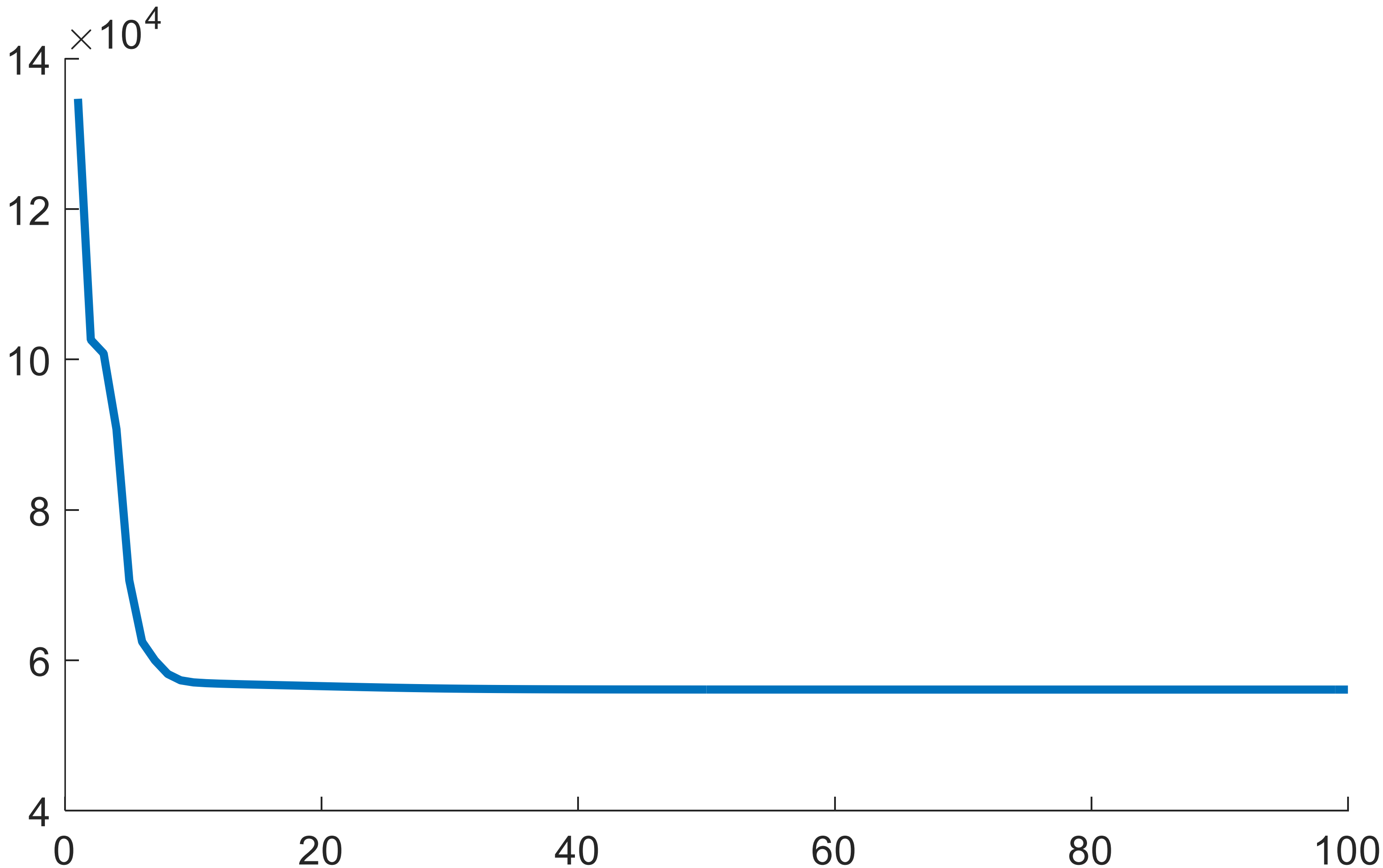
分3类结果



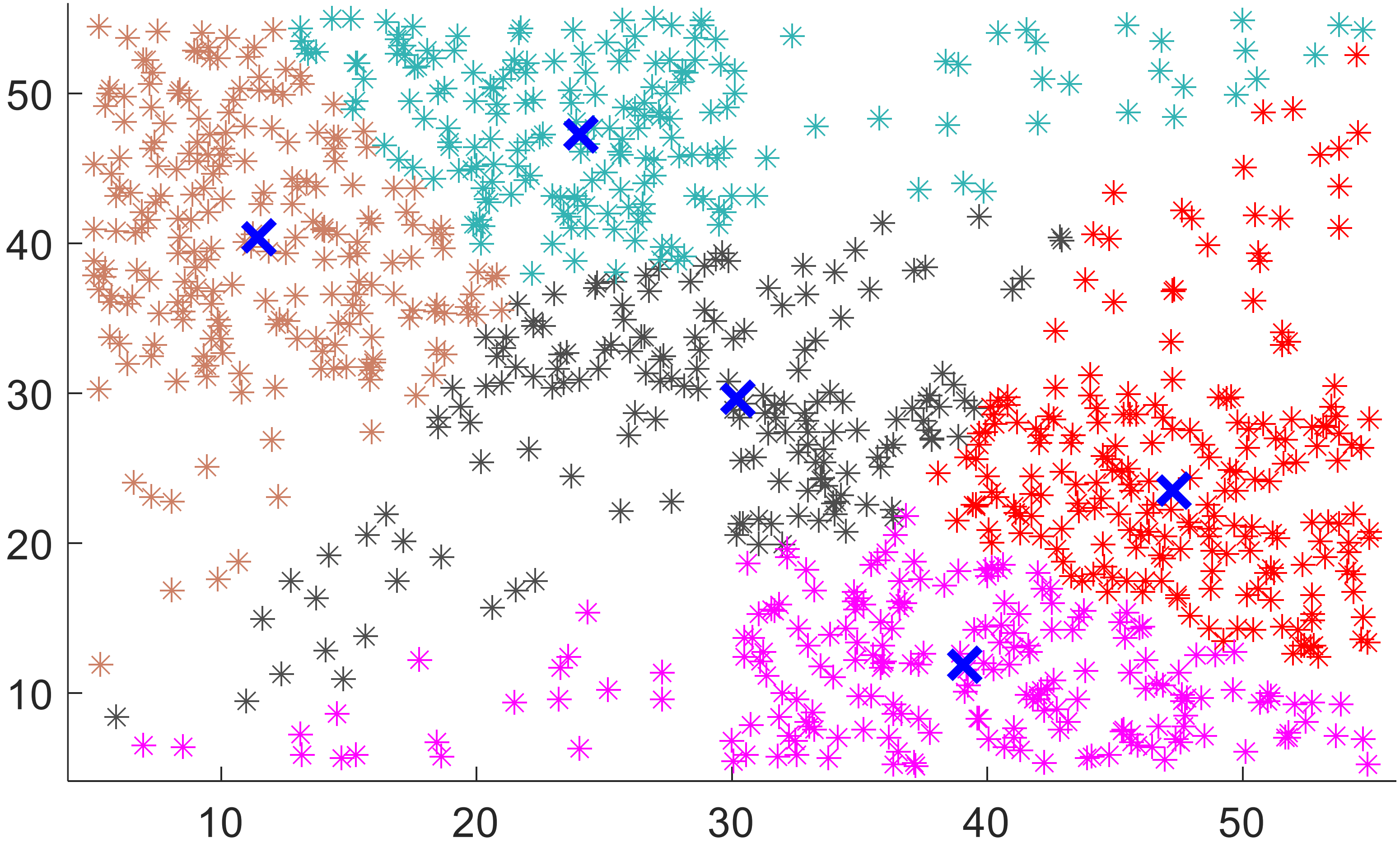
分4类结果



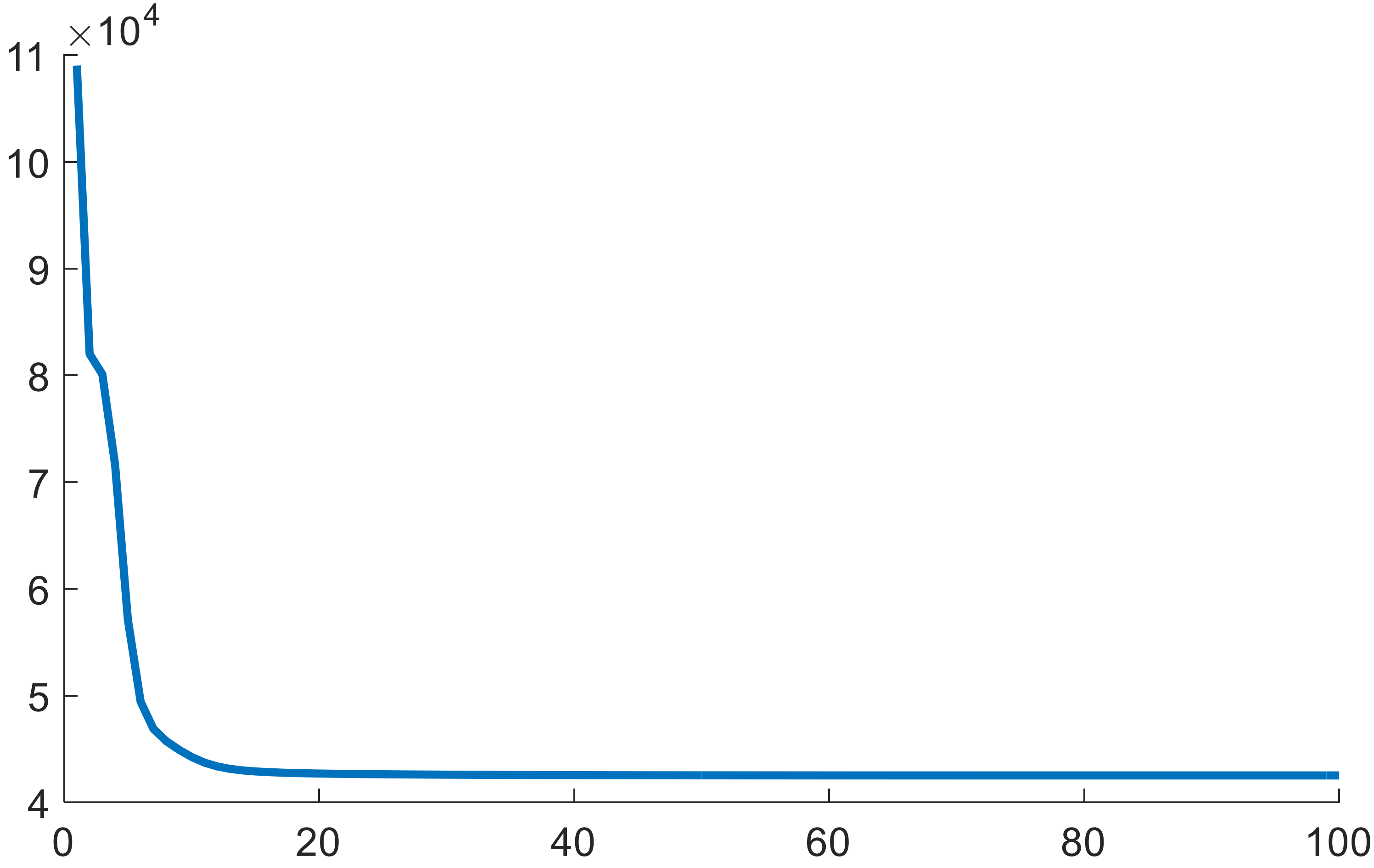
分4类结果



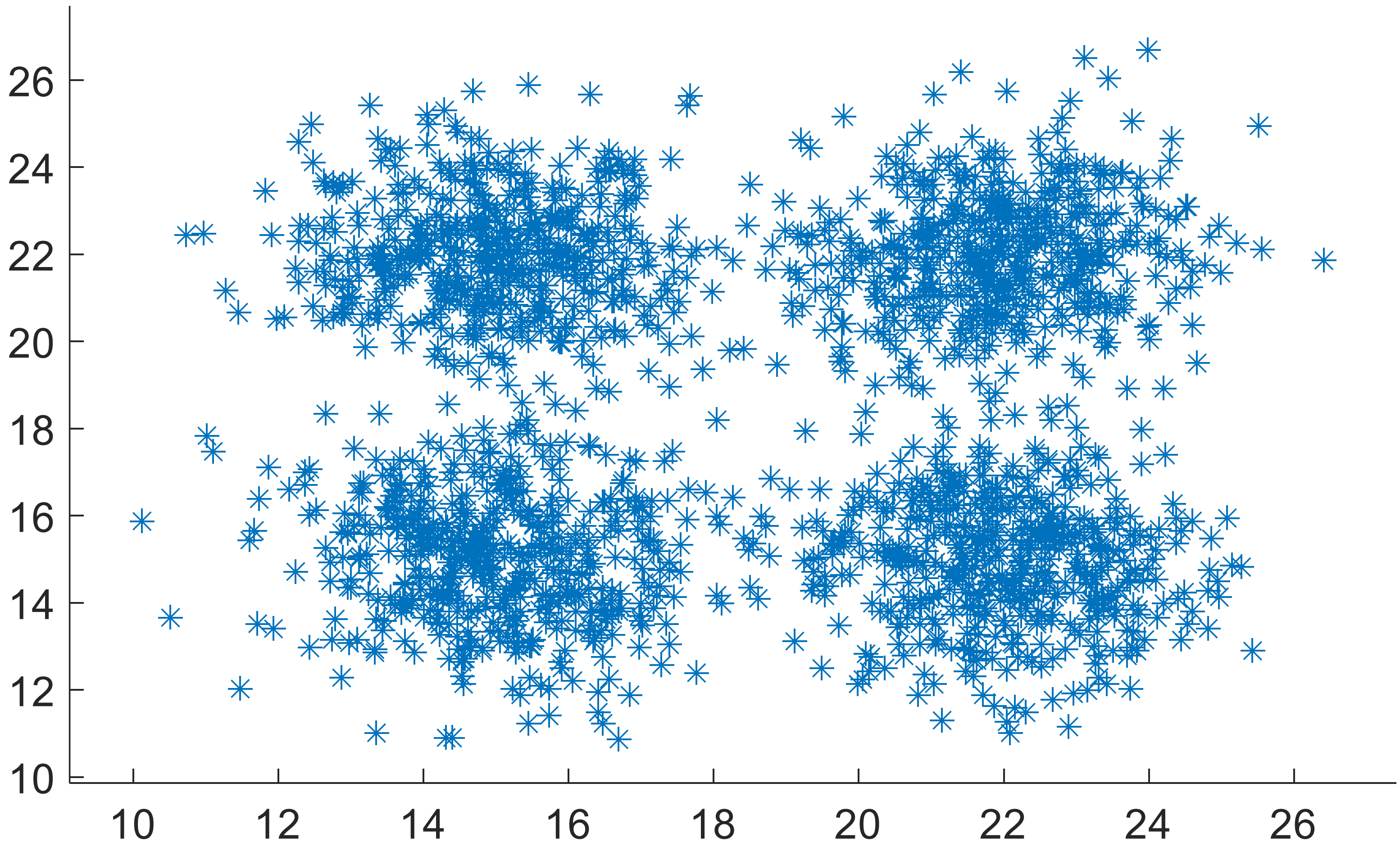
分5类结果



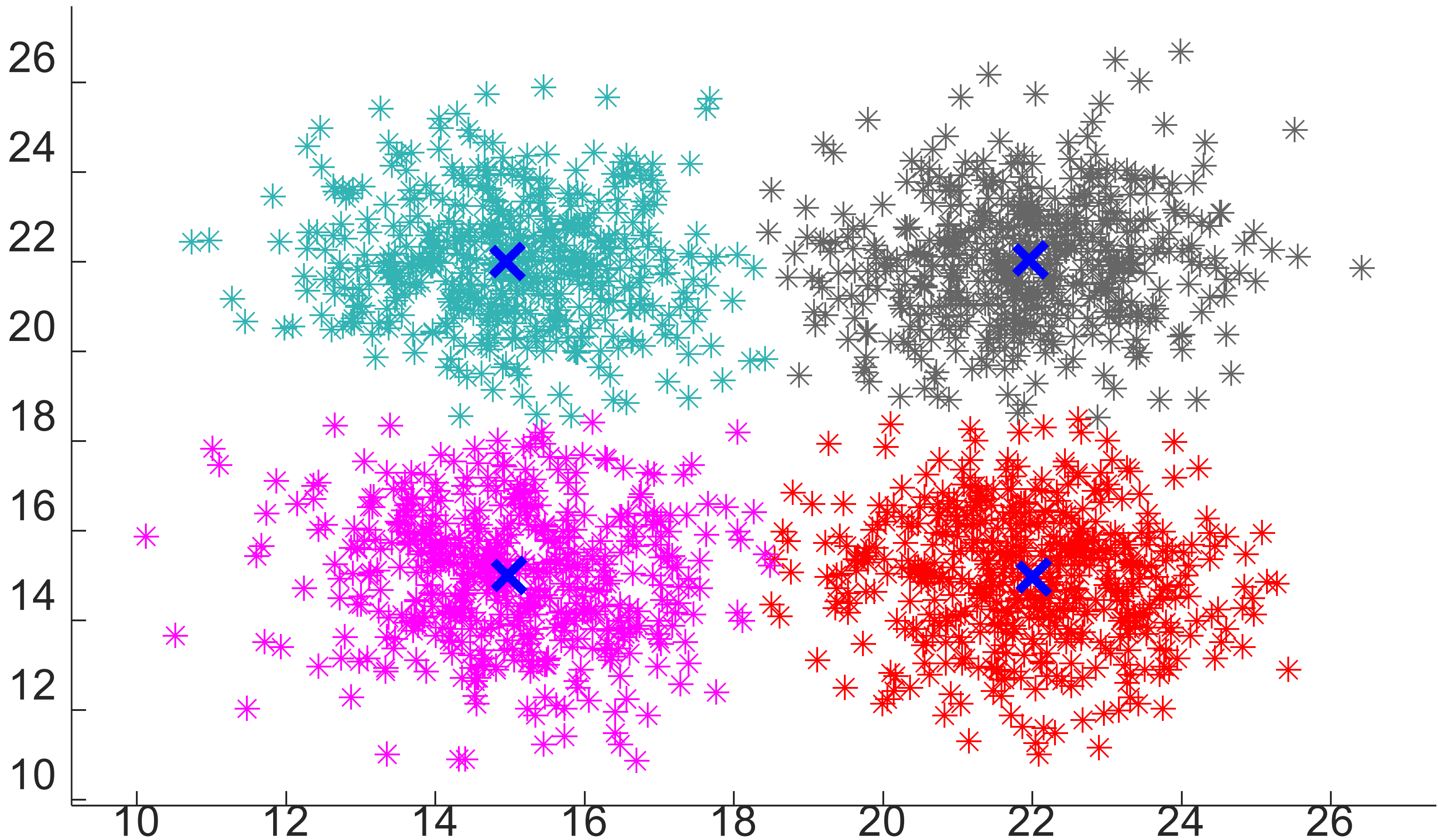
分5类结果



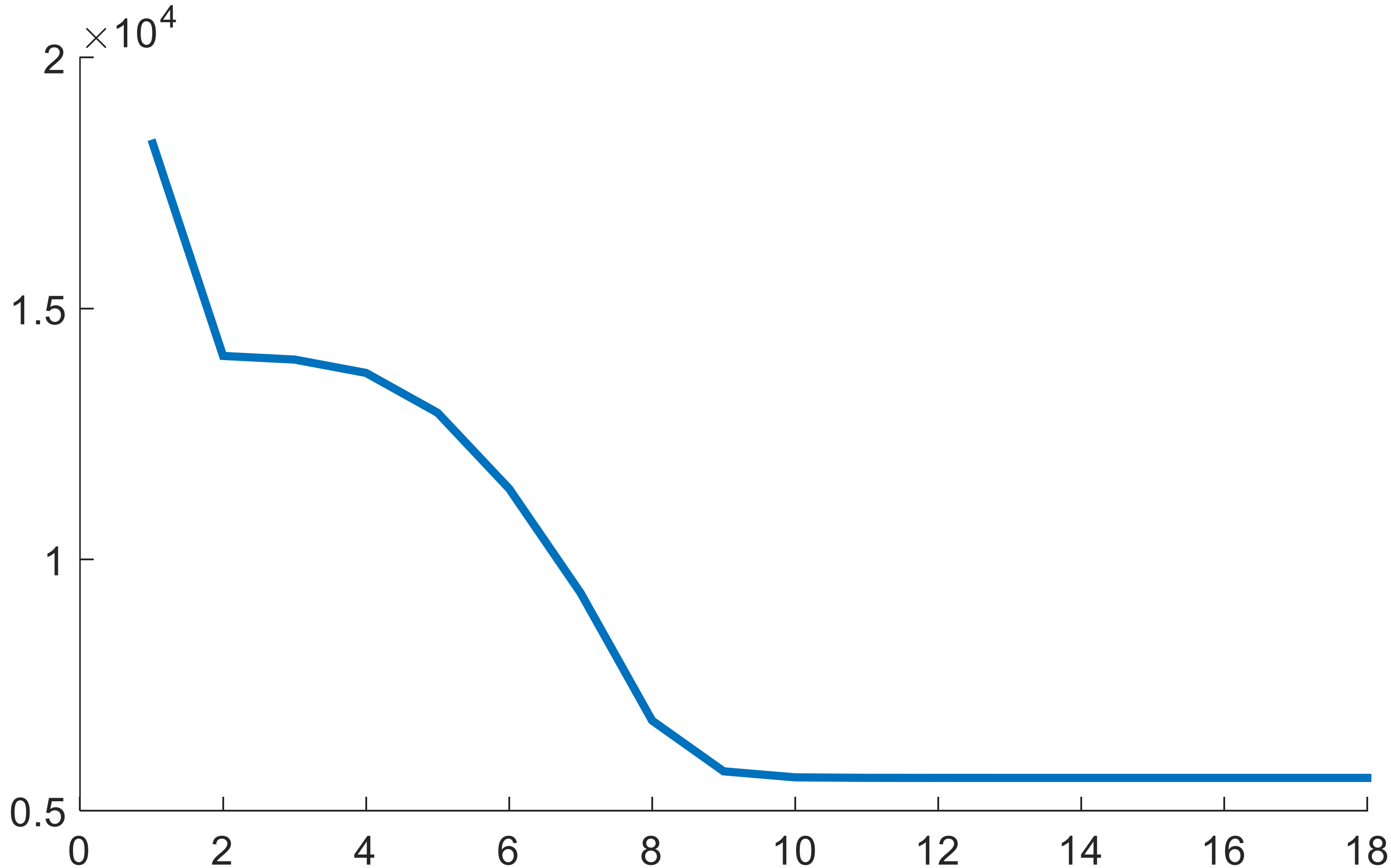
正态分布的数据



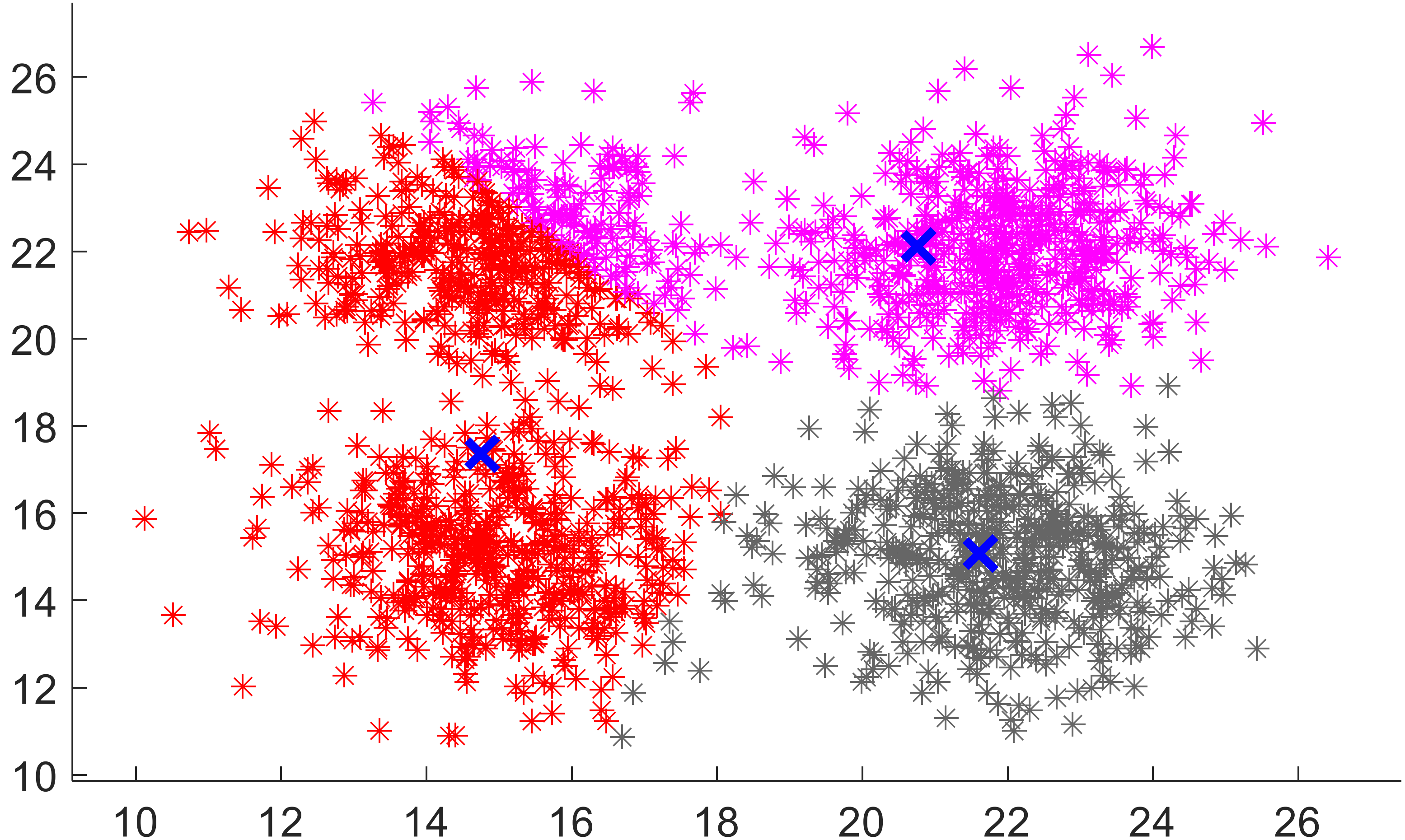
正态分布的数据分4类结果



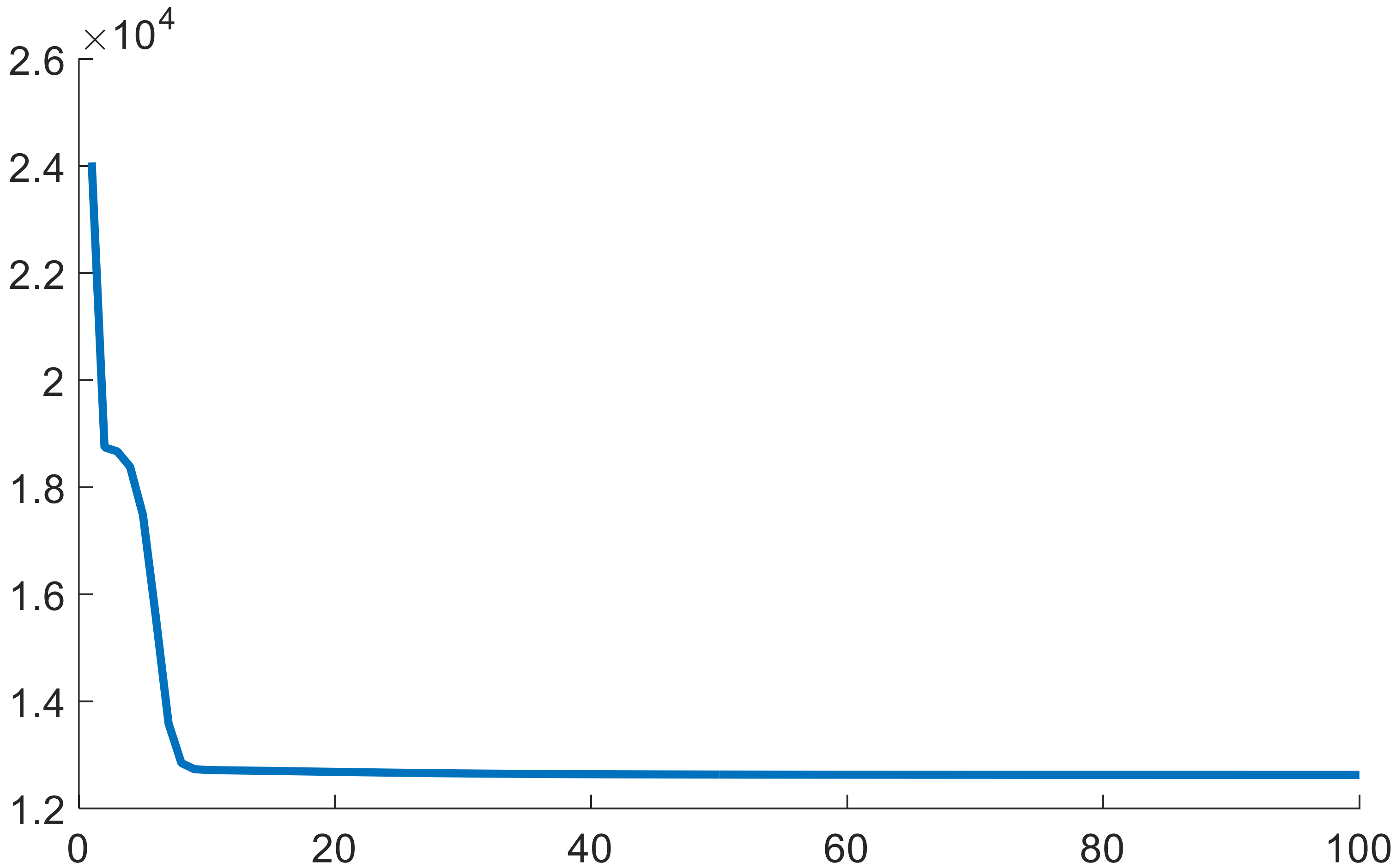
正态分布的数据分4类结果



正态分布的数据分3类结果



正态分布的数据分3类结果



算法讨论

- FCM算法是一种收敛算法，而随着数据数量、分布复杂度和聚类数量的增加，同等条件下的收敛速率也可能会变慢（并非绝对）。
- 对于正态分布的数据来说， FCM算法收敛快，且分类效果良好；而对于无规律分布的孤立点（也叫做野点）比较敏感，容易造成错分类。

FCM算法在不同领域的拓展

- 在图像处理领域，FCM算法需要结合图像的邻域信息不断更新，例如给目标函数后加上一项邻域信息约束，从而得到对应的聚类中心和隶属度更新公式（如FLICM等）。
- 在大数据处理领域，FCM算法需要被加速运行，同时保证其精确度不能有过多的丢失。因此要从算法层面对FCM的公式进行优化，还需要要使用并行计算的硬件（如GPU）加速运行。

THANK YOU !