实验 1 图像的贝叶斯分类

1.1 实验目的

将模式识别方法与图像处理技术相结合，掌握利用最小错分概率贝叶斯分类器进行图像分类的基本方法，通过实验加深对基本概念的理解。

1.2 实验仪器设备及软件

HP D538、MATLAB

1.3 实验原理

1.3.1 基本原理

阈值化分割算法是计算机视觉中的常用算法，对灰度图象的阈值分割就是先确定一个处于图像灰度取值范围内的灰度阈值，然后将图像中每个像素的灰度值与这个阈值相比较。并根据比较的结果将对应的像素划分为两类，灰度值大于阈值的像素划分为一类，小于阈值的划分为另一类，等于阈值的可任意划分到两类中的任何一类。此过程中，确定阈值是分割的关键。

对一般的图像进行分割处理通常对图像的灰度分布有一定的假设，或者说是基于一定的图像模型。最常用的模型可描述如下：假设图像由具有单峰灰度分布的目标和背景组成，处于目标和背景内部相邻像素间的灰度值是高度相关的，但处于目标和背景交界处两边的像素灰度值有较大差别，此时，图像的灰度直方图基本上可看作是由分别对应于目标和背景的两个单峰直方图混合构成。而且这两个分布应大小接近，且均值足够远，方差足够小，这种情况下直方图呈现较明显的双峰。类似地，如果图像中包含多个单峰灰度目标，则直方图可能呈现较明显的多峰。

上述图像模型只是理想情况，有时图像中目标和背景的灰度值有部分交错。这时如用全局阈值进行分割必然会产生一定的误差。分割误差包括将目标分为背

景和将背景分为目标两大类。 实际应用中应尽量减小错误分割的概率， 常用的一种方法为选取最优阈值。 这里所谓的最优阈值， 就是指能使误分割概率最小的分割阈值。图像的直方图可以看成是对灰度值概率分布密度函数的一种近似。 如一幅图像中只包含目标和背景两类灰度区域， 那么直方图所代表的灰度值概率密度函数可以表示为目标和背景两类灰度值概率密度函数的加权和。 如果概率密度函数形式已知，就有可能计算出使目标和背景两类误分割概率最小的最优阈值。

假设目标与背景两类像素值均服从正态分布且混有加性高斯噪声， 上述分类问题可以使用模式识别中的最小错分概率贝叶斯分类器来解决。以 p_1 与 p_2 分别表示目标与背景的灰度分布概率密度函数， P_1 与 P_2 分别表示两类的先验概率， 则图像的混合概率密度函数可用下式表示

$$p(x) = P_1 p_1(x) + P_2 p_2(x)$$

式中 p_1 和 p_2 分别为

$$p_1(x) = \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}$$

$$p_2(x) = \frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}}$$

$$P_1 + P_2 = 1$$

σ_1、 σ_2 是针对背景和目标两类区域灰度均值 μ_1 与 μ_2 的标准差。若假定目标的灰度较亮，其灰度均值为 μ_2，背景的灰度较暗，其灰度均值为 μ_1，因此有

$$\mu_1 < \mu_2$$

现若规定一门限值 T 对图像进行分割， 势必会产生将目标划分为背景和将背景划分为目标这两类错误。通过适当选择阈值 T，可令这两类错误概率为最小，则该阈值 T 即为最佳阈值。

把目标错分为背景的概率可表示为

$$E_1(T) = \int_{-\infty}^{T} p_2(x)\,dx$$

把背景错分为目标的概率可表示为

$$E_2(T) = \int_T^{+\infty} p_1(x)\,dx$$

总的误差概率为

$$E(T) = P_2 E_1(T) + P_1 E_2(T)$$

为求得使误差概率最小的阈值 T，可将 $E(T)$ 对 T 求导并令导数为零，可得

$$P_1 p_1(T) = P_2\, p_2(T)$$

代换后，可得

$$\ln\frac{P_1\sigma_2}{P_2\sigma_1} - \frac{(T-\mu_1)^2}{2\sigma_1^2} = -\frac{(T-\mu_2)^2}{2\sigma_1^2}$$

此时，若设 $\sigma_1 = \sigma_2 = \sigma$，则有

$$T = \frac{\mu_1 + \mu_2}{2} + \frac{\sigma^2}{\mu_1 - \mu_2}\ln\left(\frac{P_2}{P_1}\right)$$

若还有 $P_1 = P_2$ 的条件，则

$$T = \frac{\mu_1 + \mu_2}{2}$$

这时的最优阈值就是两类区域灰度均值 μ_1 与 μ_2 的平均值。

上面的推导是针对图像灰度值服从正态分布时的情况，如果灰度值服从其它分布，依理也可求出最优阈值来。一般情况下，在不清楚灰度值分布时，通常可假定灰度值服从正态分布。因此，本课题中亦可使用此方法来求得最优阈值，来对实验图像进行分割。

1.3.2 最优阈值的迭代算法

在实际使用最优阈值进行分割的过程中，需要利用迭代算法来求得最优阈值。设有一幅数字图像 $f(x,y)$，混有加性高斯噪声，可表示为

$$g(x,y) = f(x,y) + n(x,y)$$

此处假设图像上各点的噪声相互独立，且具有零均值，如果通过阈值分割将图像分为目标与背景两部分，则每一部分仍然有噪声点随机作用于其上，于是，

目标 $g_1(x, y)$ 和 $g_2(x, y)$ 可表示为

$$g_1(x, y) = f_1(x, y) + n(x, y)$$

$$g_2(x, y) = f_2(x, y) + n(x, y)$$

迭代过程中，会多次地对 $g_1(x, y)$ 和 $g_2(x, y)$ 求均值，则

$$E\{g_1(x, y)\} = E\{f_1(x, y) + n(x, y)\} = E\{f_1(x, y)\}$$

$$E\{g_2(x, y)\} = E\{f_2(x, y) + n(x, y)\} = E\{f_2(x, y)\}$$

可见，随着迭代次数的增加，目标和背景的平均灰度都趋向于真实值。因此，用迭代算法求得的最佳阈值不受噪声干扰的影响。

利用最优阈值对实验图像进行分割的迭代步骤为：

（1）确定一个初始阈值 T_0，T_0 可取为

$$T_0 = \frac{S_{min} + S_{max}}{2}$$

式中，S_{min} 和 S_{max} 为图像灰度的最小值和最大值。

（2）利用第 k 次迭代得到的阈值将图像分为目标 R_1 和背景 R_2 两大区域，其中

$$R_1 = \{ f(x, y) \mid f(x, y) \geq T_k \}$$

$$R_2 = \{ f(x, y) \mid 0 < f(x, y) < T_k \}$$

（3）计算区域 R_1 和 R_2 的灰度均值 S_1 和 S_2。

（4）计算新的阈值 T_{k+1}，其中

$$T_{k+1} = \frac{S_1 + S_2}{2}$$

（5）如果 $|T_{k+1} - T_k|$ 小于允许的误差，则结束，否则 $k = k + 1$，转步骤（2）。

利用迭代法求得最优阈值后，仍需进行一些人工调整才能将此阈值用于实验图像的分割，这是因为，这种最优阈值仍然属于全局阈值，它利用了图像中所有像素点的信息，但当光照不均匀时，图像中部分区域的灰度值可能差距较大，造成计算出的最优阈值分割效果不理想，此时，可设一人工经验因子进行校正。

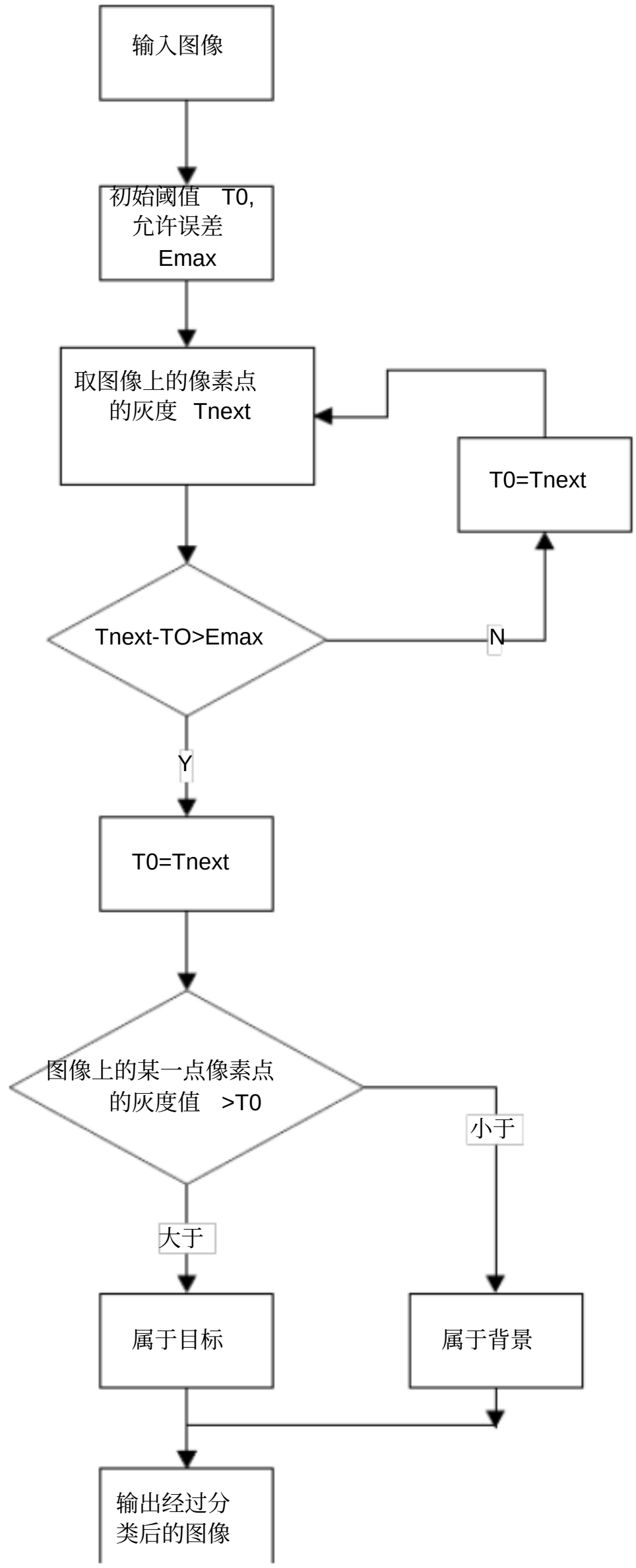
四、实验步骤及程序

1、实验步骤：

(1) 利用最优阈值对实验图像进行分割的迭代步骤编写程序流程图；

(2) 编写程序，用 Matla b 语言实现此算法，完成选择图像的分割。理解最优阈值迭代算法，设计程序实现对自选图像的最优阈值分割。

2、程序流程图：



3、程序如下：

```
a=imread(      'e:/lena.bmp'            );
figure(1)
imshow(a)
b=a(:);
Smax=max(b);
Smin=min(b);
T0=(Smax+Smin)/2;
delta=1;
while     delta>=0.1
   clear          R1   R2
   m=1;
   l=1;
      for    n=1:65536;
             if    b(n)<=T0
         R1(m)=b(n);
         m=m+1;
             else
         R2(l)=b(n);
         l=l+1;
             end
      end
   Tnext=0.5*(mean(R1)+mean(R2));
   delta=abs(Tnext-T0);
   T0=Tnext;
end
for    n=1:256
       for    m=1:256
              if    a(n,m)>=T0
          c(n,m)=1;
              else
          c(n,m)=0;
              end
       end
end
figure(2)
imshow(c)
```

五、实验结果与分析

分割域值：T0 =124.3653

分割前原图：

分割后效果图：

结论：利用迭代法求得最优阈值仍然属于全局阈值。　由图可知分割后的效果图有两个灰度级，即分为目标背景两个部分。 将原图中的每一个像素与　Tnest作比较，大于的为目标，小于的为背景。

实验 2 K 均值聚类算法

2.1 实验目的

将模式识别方法与图像处理技术相结合， 掌握利用 K 均值聚类算法进行图像分类的基本方法，通过实验加深对基本概念的理解。

2.2 实验仪器设备及软件

HP D538、MATLAB 、 WIT

2.3 实验原理

K 均值聚类法分为如下几个步骤：

一、初始化聚类中心

1、凭经验选择。根据具体问题，凭经验从样本集中选出 K 个比较合适的样本作为初始聚类中心。

2、用前 K 个样本作为初始聚类中心。

3、将全部样本随机地分成 K 类，计算每类的样本均值，将样本均值作为初始聚类中心。

4、密度法。以每个样本为球心，用某个正数为半径作一个球形邻域，落在邻域内的样本数为该点密度， 选密度最大点为第一初始聚类中心。 在离开第一点规定距离范围外确定次大密度点，以避免初始聚类中心聚集。

5、从 K −1 个聚类划分的解中产生 K 个聚类划分的初始聚类中心。 先把全部样本看作一个聚类， 其聚类中心为样本的总均值； 然后确定两聚类问题的聚类中心是一聚类问题的总均值和离它最远的点；以此类推。

二、初始聚类

1、按就近原则将样本归入各聚类中心所代表的类中。

2、取一样本，将其归入与其最近的聚类中心的那一类中，重新计算样本均值，更新聚类中心。然后取下一样本，重复操作，直至所有样本归入相应类中。

三、判断聚类是否合理

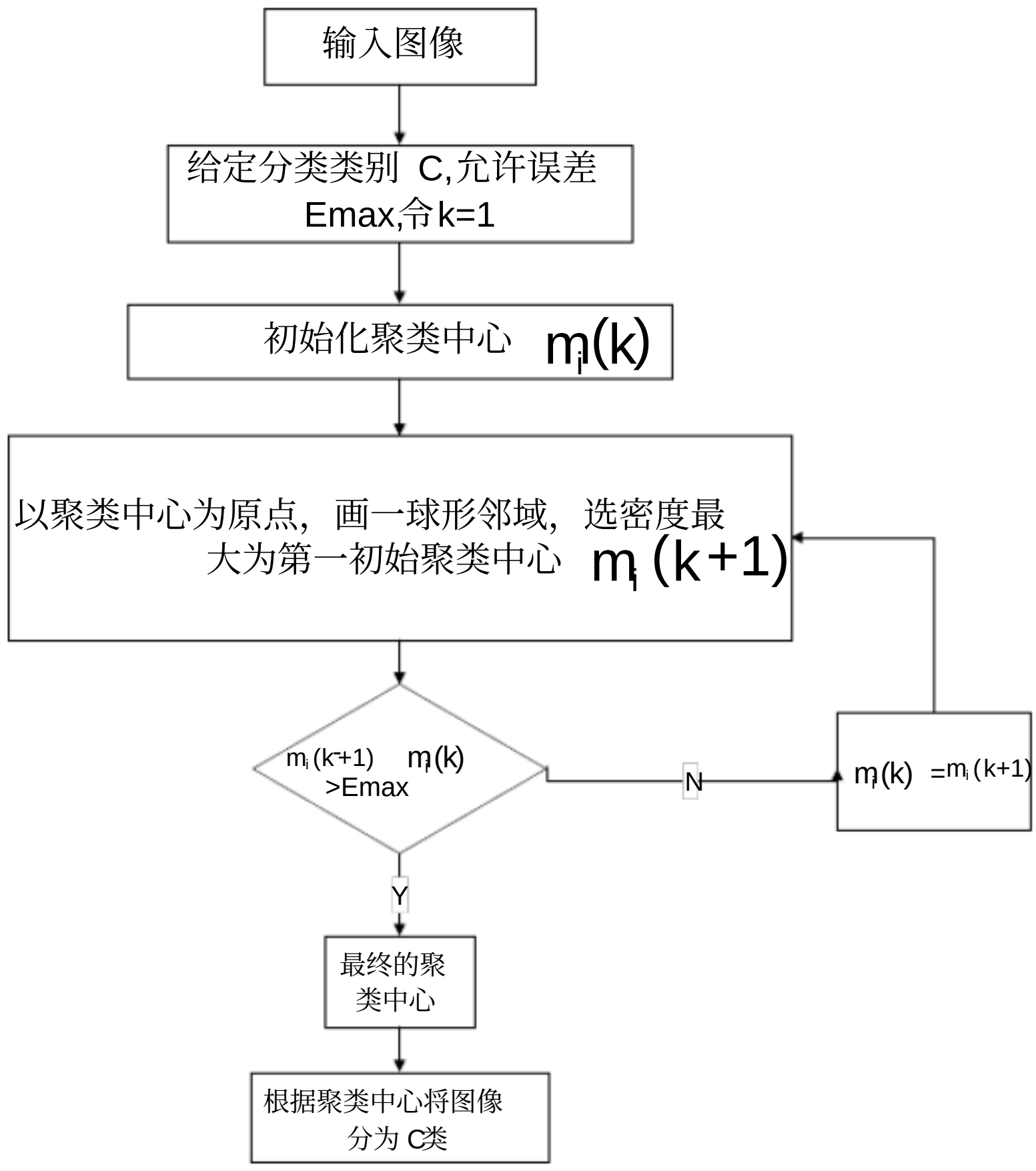
采用误差平方和准则函数判断聚类是否合理，　　不合理则修改分类。 循环进行判断、修改直至达到算法终止条件。

2.4 实验步骤及程序

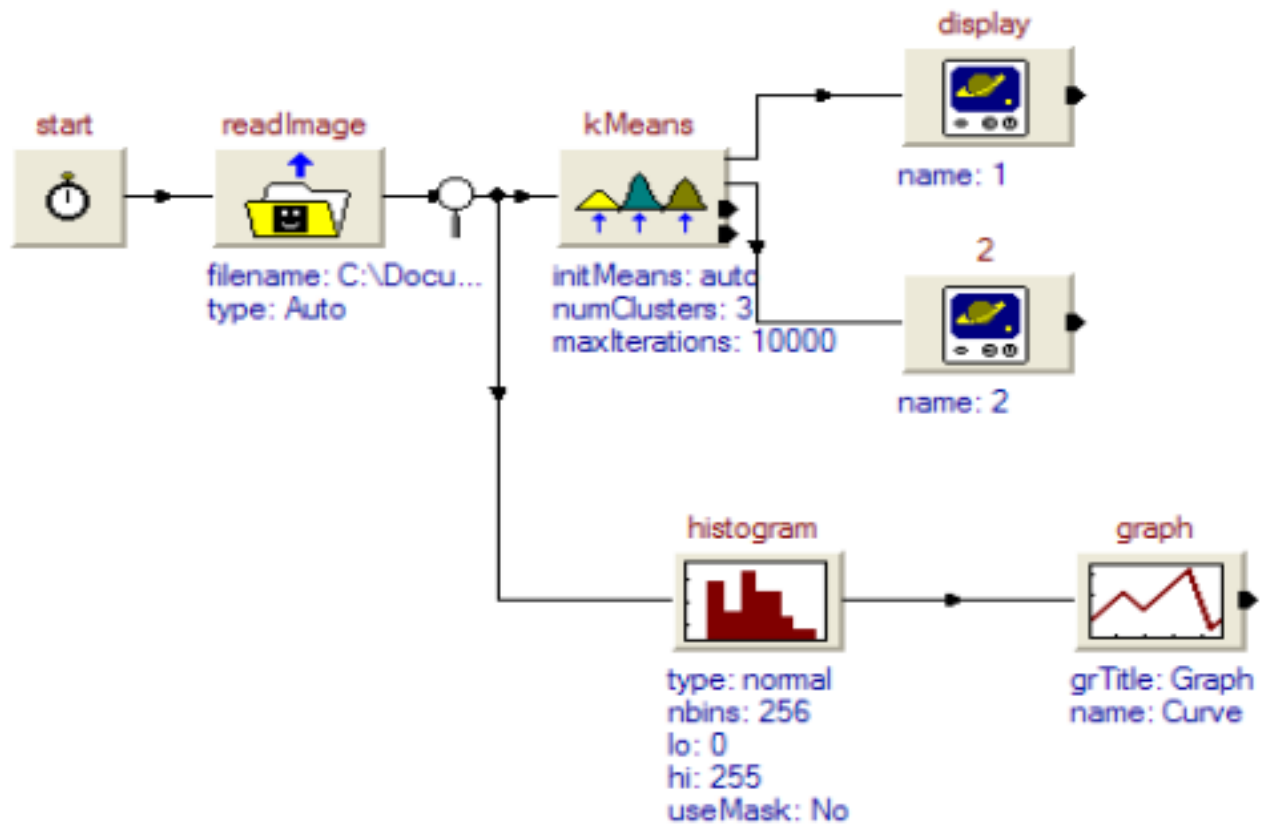
1、实验步骤

(1) 利用 K-均值聚类法的算法步骤编写程序流程图；
(2) 编写程序，用 Matlab 语言实现此算法，完成选择图像的分割。
(3) 利用 WIT 实现 K 均值聚类算法的图像分割

2、程序流程图



WTI聚类程序如图：



3、程序如下：

```
tic
img= imread (          'e:/lena.bmp'          );
[m,n]=size(img);
subplot(2,2,1),imshow(img);title(                    ' 原始图 ' )
subplot(2,2,2),imhist(img);title(                    ' 灰度直方图 ' )
hold      off    ;
img=double(img);
for     i=1:200
    c1(1)=25;
    c2(1)=125;
    c3(1)=200;
    r=abs(img-c1(i));
    g=abs(img-c2(i));
    b=abs(img-c3(i));
    r_g=r-g;
    g_b=g-b;
    r_b=r-b;
    n_r=find(r_g<=0&r_b<=0);
    n_g=find(r_g>0&g_b<=0);
        n_b=find(g_b>0&r_b>0);
        i=i+1;
   c1(i)=sum(img(n_r))/length(n_r);
c2(i)=sum(img(n_g))/length(n_g);    c3(i)=sum(img(n_b))/length(n_b);
d1(i)=(c1(i)-c1(i-1))^2;
    d2(i)=(c2(i)-c2(i-1))^2;
    d3(i)=(c3(i)-c3(i-1))^2;
        if    d1(i)<=0.001&&d2(i)<=0.001&&d3(i)<=0.001
        R=c1(i);
```

```
        G=c2(i);
        B=c3(i);
        k=i;
              break    ;
        end
end
R
G
B
img=uint8(img);
img(find(img<R))=0;
img(find(img>R&img<G))=128;
img(find(img>G))=255;
toc
subplot(2,2,3),imshow(img);title(                    ' 聚类后图象 ' )
subplot(2,2,4),imhist(img);title(                    ' 聚类后灰度直方图  ' )
```

2.5 实验结果与分析

1、MATLAB 实验结果：

聚类类别数： K=3

聚类中心：

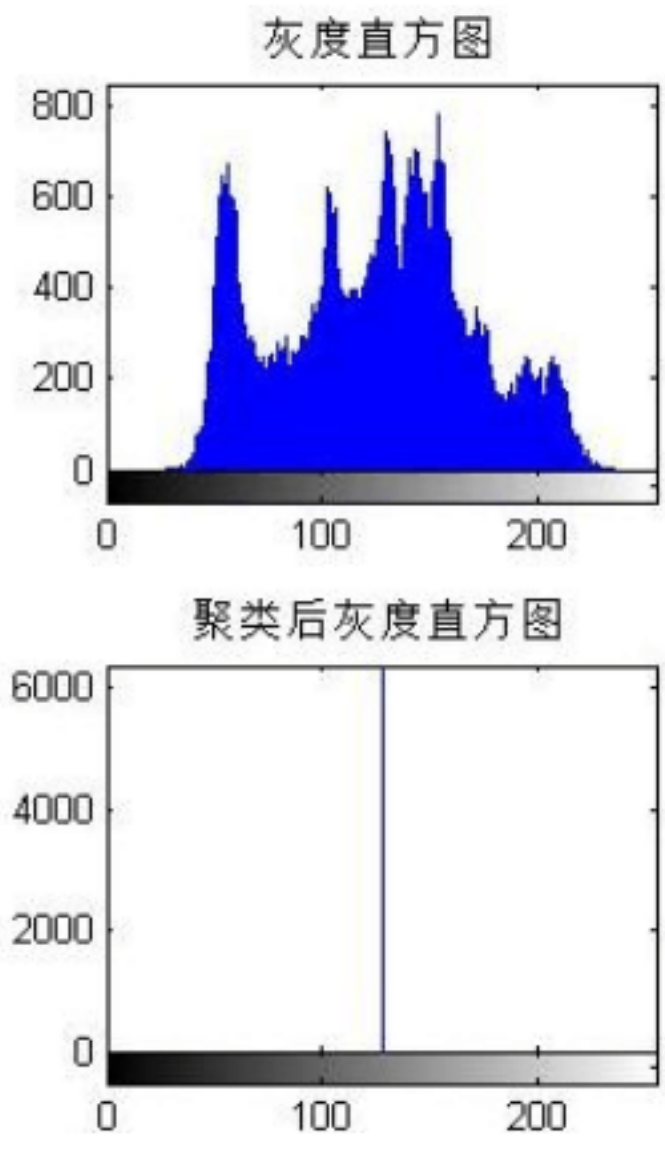
R =68.3649

G =129.3456

B =181.5449

运行时间： Elapsed time is 3.042837 seconds.

迭代次数： n =256



2、WIT 实验结果：

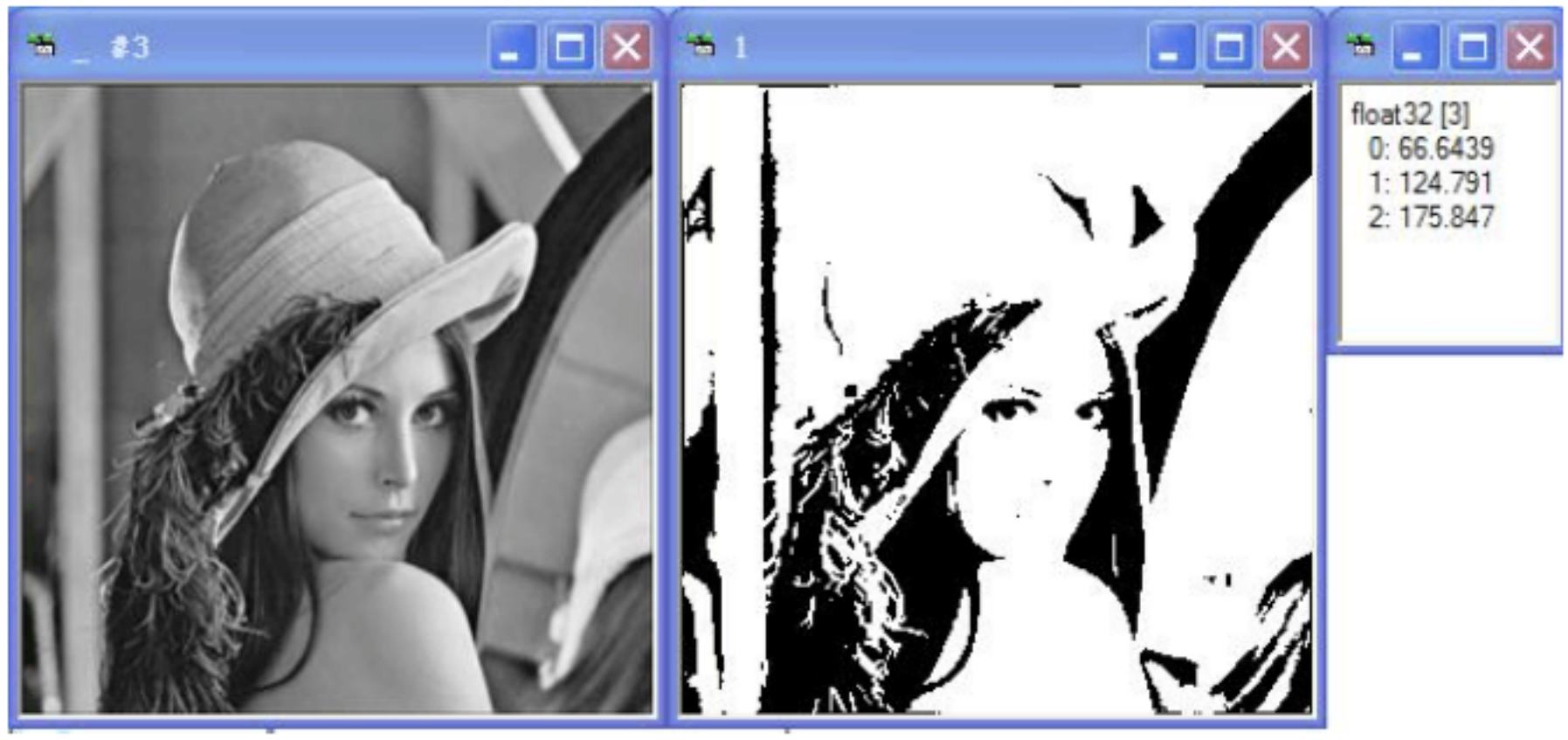
聚类类别数： K=3

聚类中心： R =66.6439

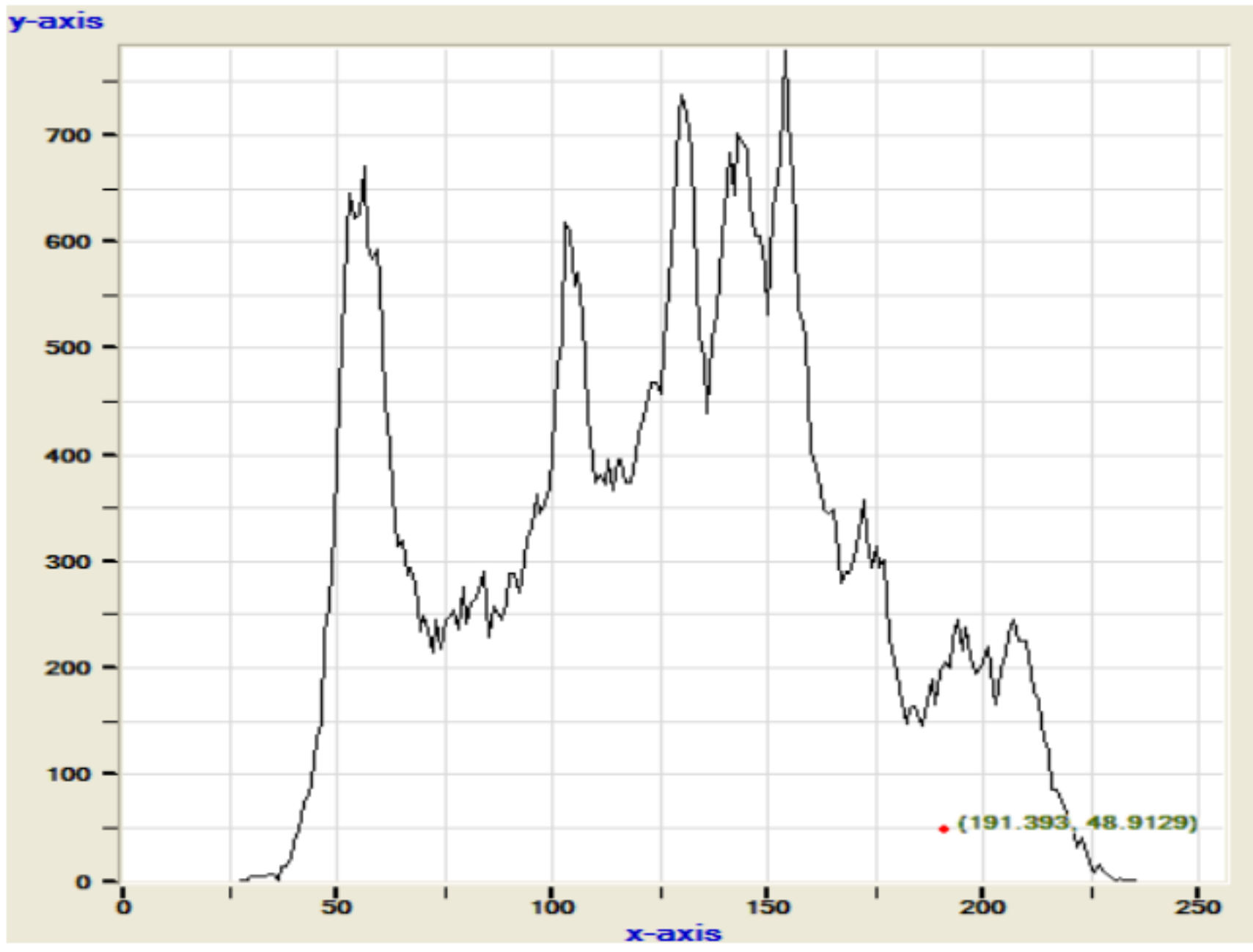
G =124.791

B =175.847

运行时间： Elapsed time is 2.359378 seconds



灰度直方图：



结论：两种实验结果所得聚类中心相近，说明基于 K-均值算法利用 matlab 编写的图像分割程序是有效的。

实验 3 神经网络模式识别

3.1 实验目的

掌握利用感知器和 BP 网进行模式识别的基本方法，通过实验加深对基本概念的理解。

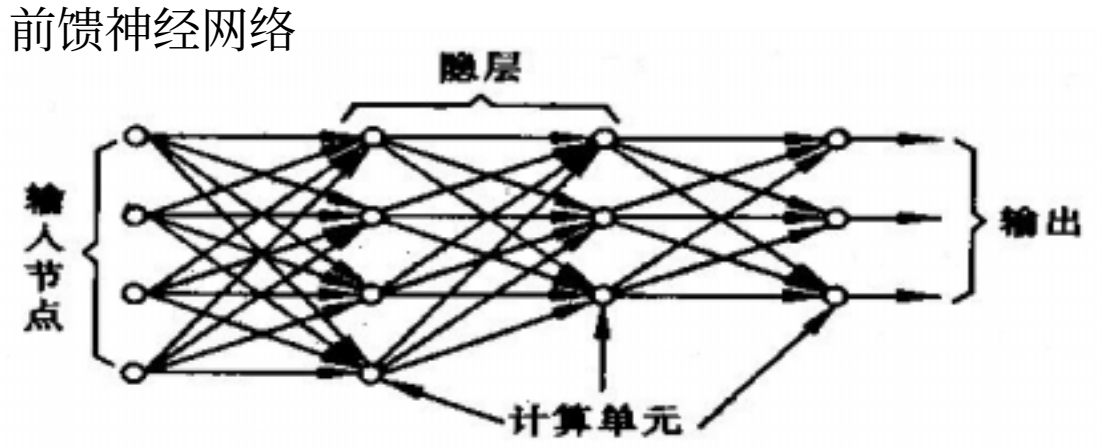
3.2 实验仪器与设备

HP D538、 MATLAB

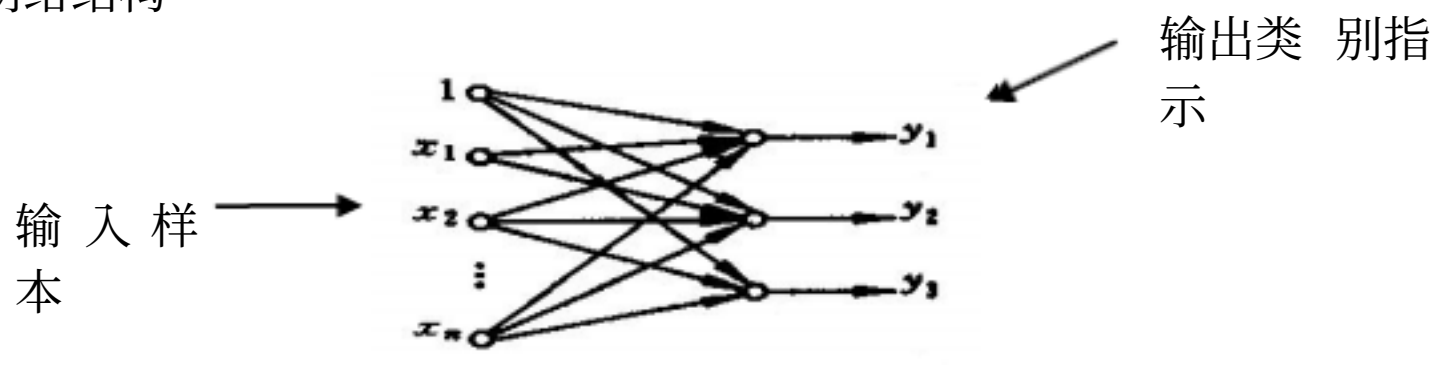
3.3 实验原理

感知器原理

前馈神经网络



感知器网络结构



$$\omega_{ij}(t+1) = \omega_{ij}(t) + \Delta\omega_{ij}(t)$$

$$\Delta\omega_{ij} = \eta(y_j - \hat{y}_j)x_i$$

单层神经网络，只能解决线性可分问题。

神经网络特点

分布式存储信息，用神经网络间连接权值的分布来表示特定的信息，当局部网络受损，仍能恢复

原来的信息。

对信息的处理具有并行性。 每个神经元都可以根据接收到的信息作独立的运算和处理，然后将结

果传输出去，体现了并行处理的概念。

对信息的处理具有自组织、自学习的特点。通过改变连接权值适应周围环境变化，称为神经元学习过程。

BP 法原理

一般为两层前馈神经网络，激励函数为 Sigmoid 函数。

基本思想：根据样本希望输出与实际输出之间的平方误差最小，利用梯度下降法，从输出层开始，

逐层修正权系数。修正周期分两个阶段：前向传播阶段，反向传播阶段。

正向过程：

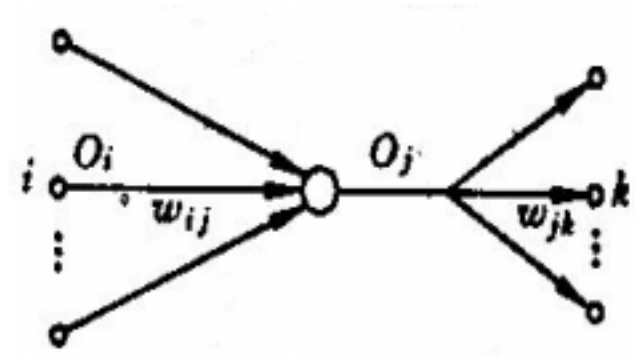
$O_i = x_i, net_j = \sum_i \omega_{ij} O_i$

$O_j = f(net_j) = \frac{1}{1+e^{-net_j}}$

备注：$f'(net_j) = f(net_j)[1 - f(net_j)]$

$net_k = \sum_j \omega_{jk} O_j$

$O_k = f(net_k) = \hat{y}_k$ 输出



反向一层：$E = \frac{1}{2}\sum_k (y_k - \hat{y}_k)^2$

梯度下降：$\omega_{jk}(t+1) = \omega_{jk}(t) + \Delta\omega_{jk}$

$\delta_k = \frac{\partial E}{\partial net_k}$

$\frac{\partial E}{\partial \omega_{jk}} = \frac{\partial E}{\partial net_k}\frac{\partial net_k}{\partial \omega_{jk}} = \delta_k O_j$

$\Delta\omega_{jk} = -\eta\delta_k O_j$

$\delta_k = \frac{\partial E}{\partial \hat{y}_k}\frac{\partial \hat{y}_j}{\partial net_k} = -(y_k - \hat{y}_k) f'(net_k)$

$= -(y_k - \hat{y}_k)\hat{y}_k(1 - \hat{y}_k)$

反向二层：$\omega_{ij}(t+1) = \omega_{ij}(t) + \Delta\omega_{ij}$

$\Delta\omega_{ij} = -\eta\delta_j O_i$

$\delta_j = \frac{\partial E}{\partial net_j} = \sum_k \frac{\partial E}{\partial net_k}\frac{\partial net_k}{\partial O_j}\frac{\partial O_j}{\partial net_j}$

$= \sum_k \delta_k \omega_{jk} f'(net_j)$

$= \sum_k \delta_k \omega_{jk} O_j(1 - O_j)$

BP 采用 S 函数，输出不宜设为 1 或 0，可设为 0.9 或 0.1。

权系数初始化：不应将初始值设为相同，否则在学习过程中始终不变，可设为随机值。

步长的选择：应设为可变步长，以防止震荡。

局部最小问题： BP 算法是非线性优化算法，初始值设置不当，可能陷入局部极小。

前馈网络结构：输入节点数为模式维数，输出节点数一般为类别数，隐层节点数尚无明确方法，实验确定。

3.4 实验步骤及程序

1、实验步骤

感知器实验： 1、设计线性可分实验，要求训练样本 10个以上

2 、奇异样本对网络训练的影响

3 、以线性不可分样本集训练分类器

BP网实验：利用 BP网对上述线性不可分样本集进行分类

2、实验程序：

（1）设计线性可分实验

设计线性分类器对线性可分样本集进行分类，样本数目 10个以上，训练及分类步骤齐全，记录分类器训练的迭代次数和训练时间。

程序如下：

```
close all;
clear;
clc;
tic;
P=[-3 -1 -5 4 2 -4 -2 1 4 3 1 -3 ;5 2 1 4 1 -1 -3 -1 -2 -4 5 -6 ]; %P为输入矢量
T=[1 1 1 1 0 1 0 0 0 0 1 0 ];                                      %T为目标矢量
figure,plotpv(P,T),title('对线性可分样本集进行分类 ')；
```

```
net=newp(minmax(P),1);                          %创建感知器神经网络  ,一个神经元
linehandle=plotpc(net.IW{1},net.b{1});                  %net.iw权值 ,net.b阈值
E=1;n=0;
while(sse(E))                                           %训练感知器神经网络
        [net,y,E]=adapt(net,P,T);                       %用新的权值建立神经网络
        n=n+1;
        perf(n)=sse(E);                                 %误差的平方和

linehandle=plotpc(net.IW{1},net.b{1},linehandle);drawnow;
                                                        %绘制分类线
    end
    toc;
    n                                                   %迭代次数
    figure,plot(perf),title(' 训练样本误差平方和  ');          %绘制误差变化曲线
```

（2）奇异样本对网络训练的影响

奇异样本：该样本向量同其他样本向量比较起来特别大或特别小时，网络训练所花费的时间将很长。设计实验考察奇异样本对感知机训练的影响，比较有无奇异点时的训练时间及迭代次数，设计解决此问题的方案并实验验证。解决方案：对样本归一化， learnpn，其对无奇异点样本集效率较低。

程序如下：

```
close all;
clear;
clc;tic;
%P 为输入矢量
P=[-3 -1 -5 4 2 -4 -2 1 4 3 1 50;5 2 1 4 1 -1 -3 -1 -2 -4 5 10];%T为目标矢量
T=[1 1 1 1 0 1 0 0 0 0 1 0 ];
figure,plotpv(P,T),title(' 分类数据点图  ');%绘制待分类数据点图
net=newp(minmax(P),1,'hardlim','learnpn');%创建感知器神经网络
linehandle=plotpc(net.IW{1},net.b{1});
E=1;n=0;
%训练感知器神经网络
while(sse(E))
        [net,y,E]=adapt(net,P,T);
        n=n+1;
        perf(n)=sse(E);
        linehandle=plotpc(net.IW{1},net.b{1},linehandle);drawnow;
    end
    toc
    n
    %绘制误差曲线
    figure,plot(perf),title(' 绘制误差变化曲线  ');%绘制误差变化曲线
```

（3）以线性不可分样本集训练分类器

使用 BP 网络能适用该样本， 该 BP 网络使用 L-M 优化算法 ,能有效减少迭代次数和训练时间，较 traingdm 效率更高。

程序如下：

```
close all;
clear;
clc;tic;
P=[-3 -1 -5 4 2 -4 -2 1 -2 3 1 -3;
    2 2 1 4 1 -1 0 -1 1 0 2 -2]; %P为输入矢量
T=[1 1 1 1 0 1 0 0 0 0 1 0;
   1 1 1 1 1 1 1 1 1 1 1 1 ];%T为目标矢量
figure,plotpv(P,T),title(' 绘制数据点图 ');%绘制待分类数据点图
net=newff(minmax(P),[9,9,2],{'tansig','tansig','purelin'},'trainlm');% 用前馈反向传播网络
inputWeights=net.IW{1,1}% 当前输入层权值和阈值
inputbias=net.b{1};
layerWeights=net.LW{2,1}% 当前网络层权值和阈值
layerbias=net.b{1};
outputWeights=net.LW{3,2}
%设置训练参数
% net.trainParam.show=50;
% net.trainParam.lr=0.05;
% net.trainParam.mc=0.9;
net.trainParam.epochs=100;%对数据组进行重复 100 次训练
net.trainParam.goal=1e-2;%误差平方和
[net,tr]=train(net,P,T);%调用 TRAINGDM 算法训练 BP 网络
toc
A=sim(net,P);%对样本和网络进行训练，得出目标矢量
A
E=T-A;
MSE=mse(E)%均方误差
```

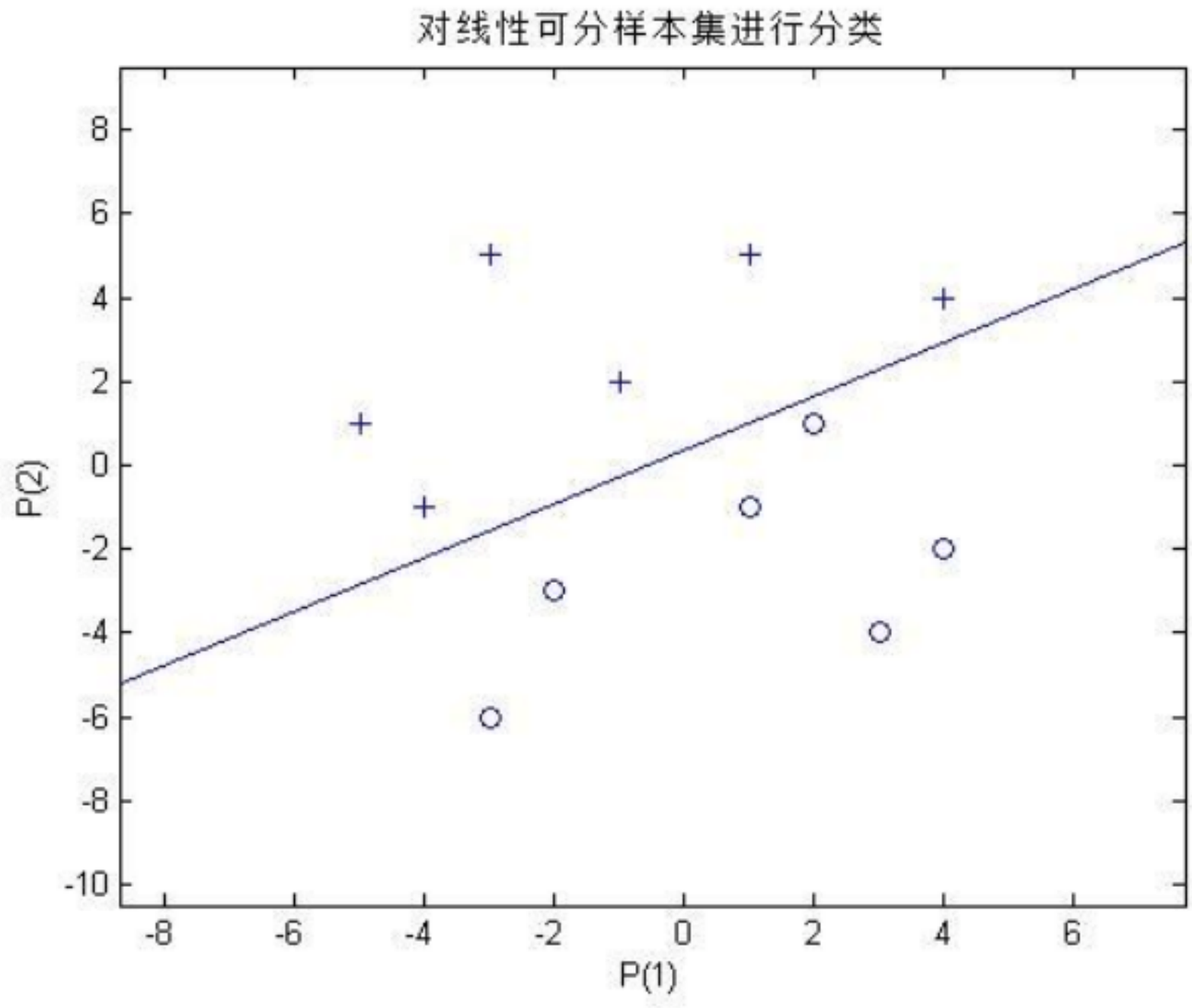
3.5 实验结果与分析

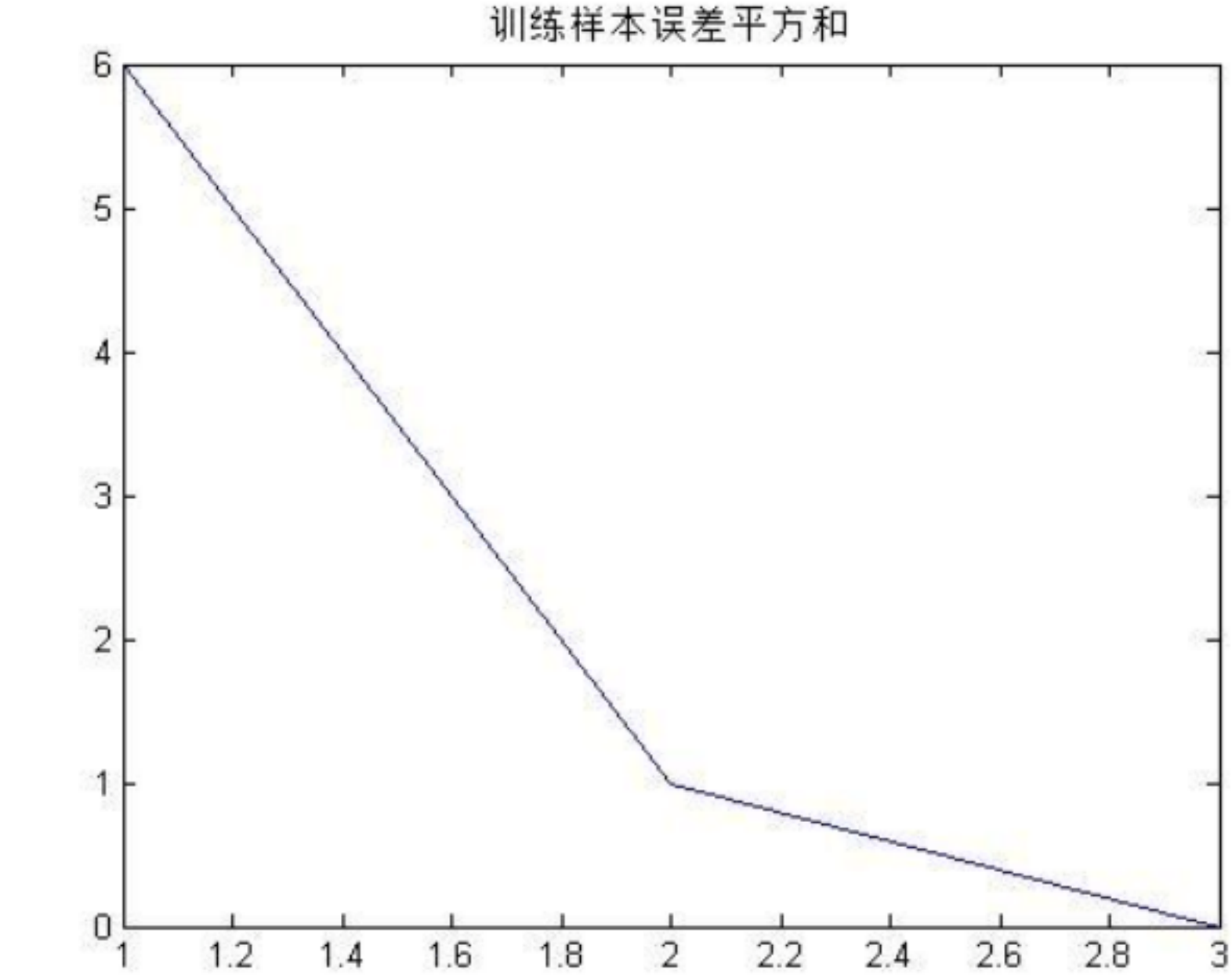
1、设计线性可分实验

运行时间： Elapsed time is 0.582415 seconds.

迭代次数： n = 3

实验截图如下：


对线性可分样本集进行分类
8
6
4
2
0
-2
-4
-6
-8
-10
P(2)
-8
-6
-4
-2
0
2
4
6
P(1)


训练样本误差平方和
6
5
4
3
2
1
0
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6
2.8
3

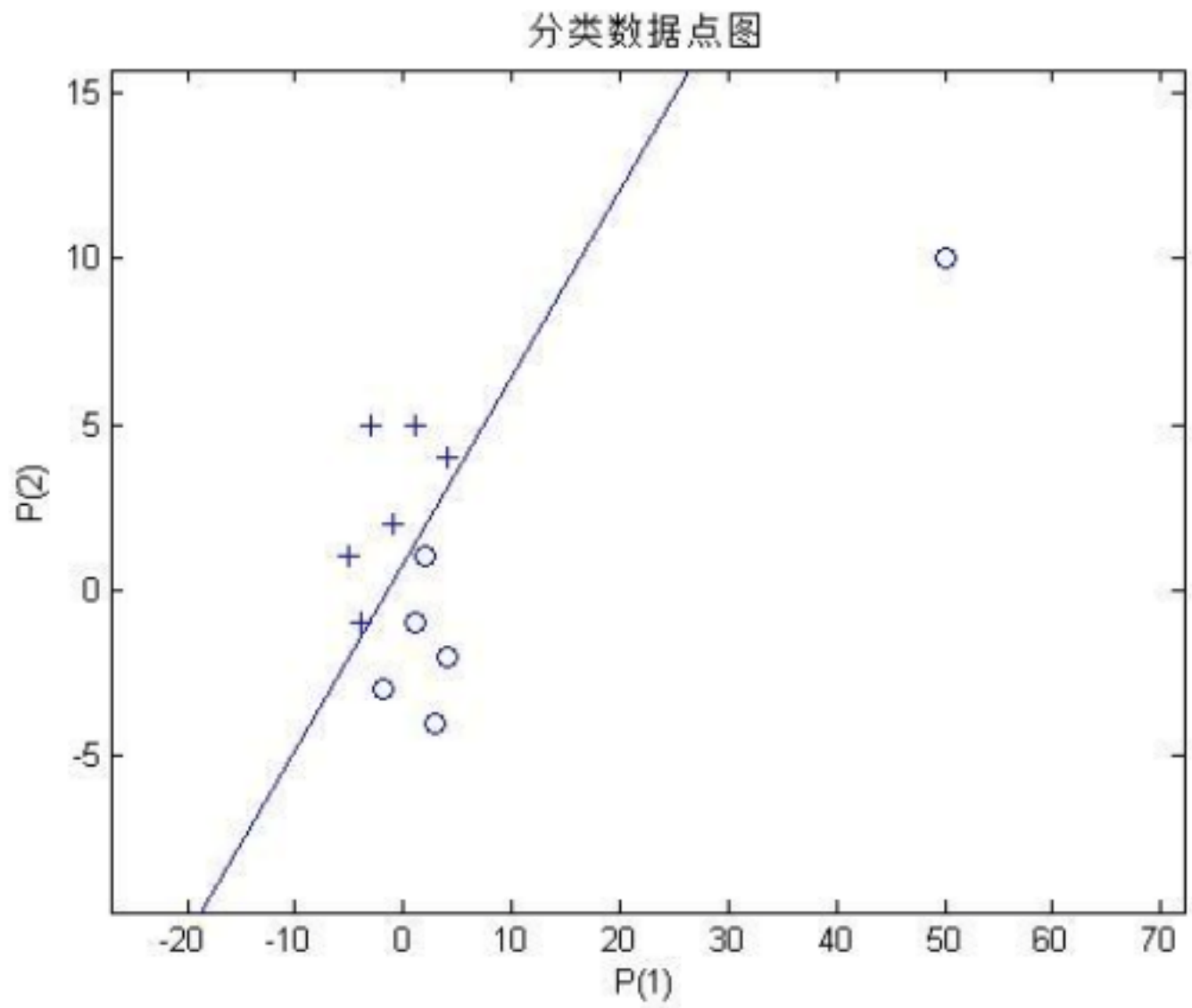
结果分析：由输入矢量建立感知器网络，通过经过网络训练得到的结果矢量与目标矢量做差，得到的误差，对其平方求和与 0 比较，若不为 0，则继续经过 adpat 算放训练建立的感知器网络，知道它能够将输入的样本集分开，并且得到的结果与目标矢量一致。此时感知器网络的阈值是一定的。得到的实验结果如图所示。

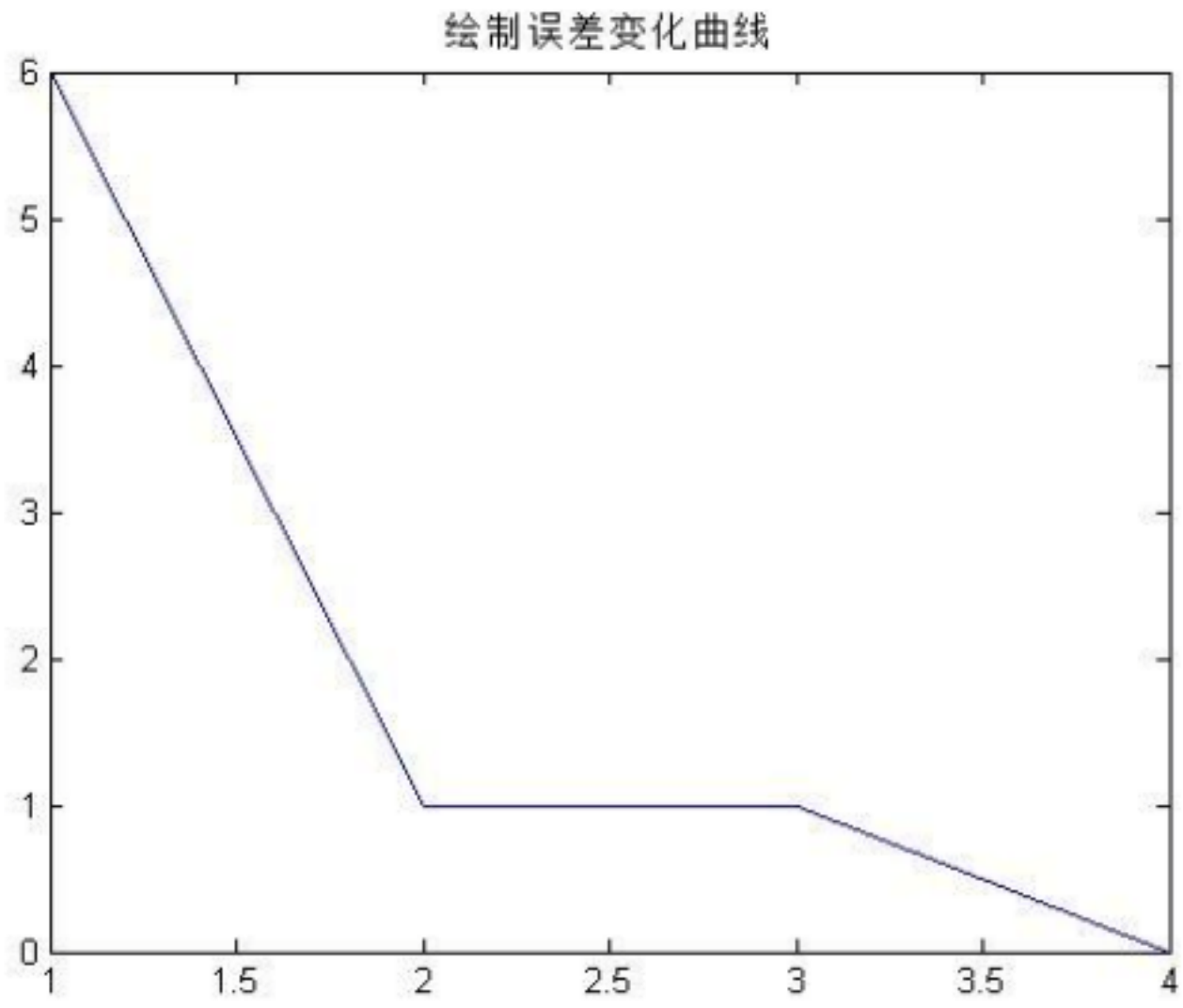
2、奇异样本对网络训练的影响

运行时间： Elapsed time is 0.721029 seconds.

迭代次数： n =4

实验截图如下：





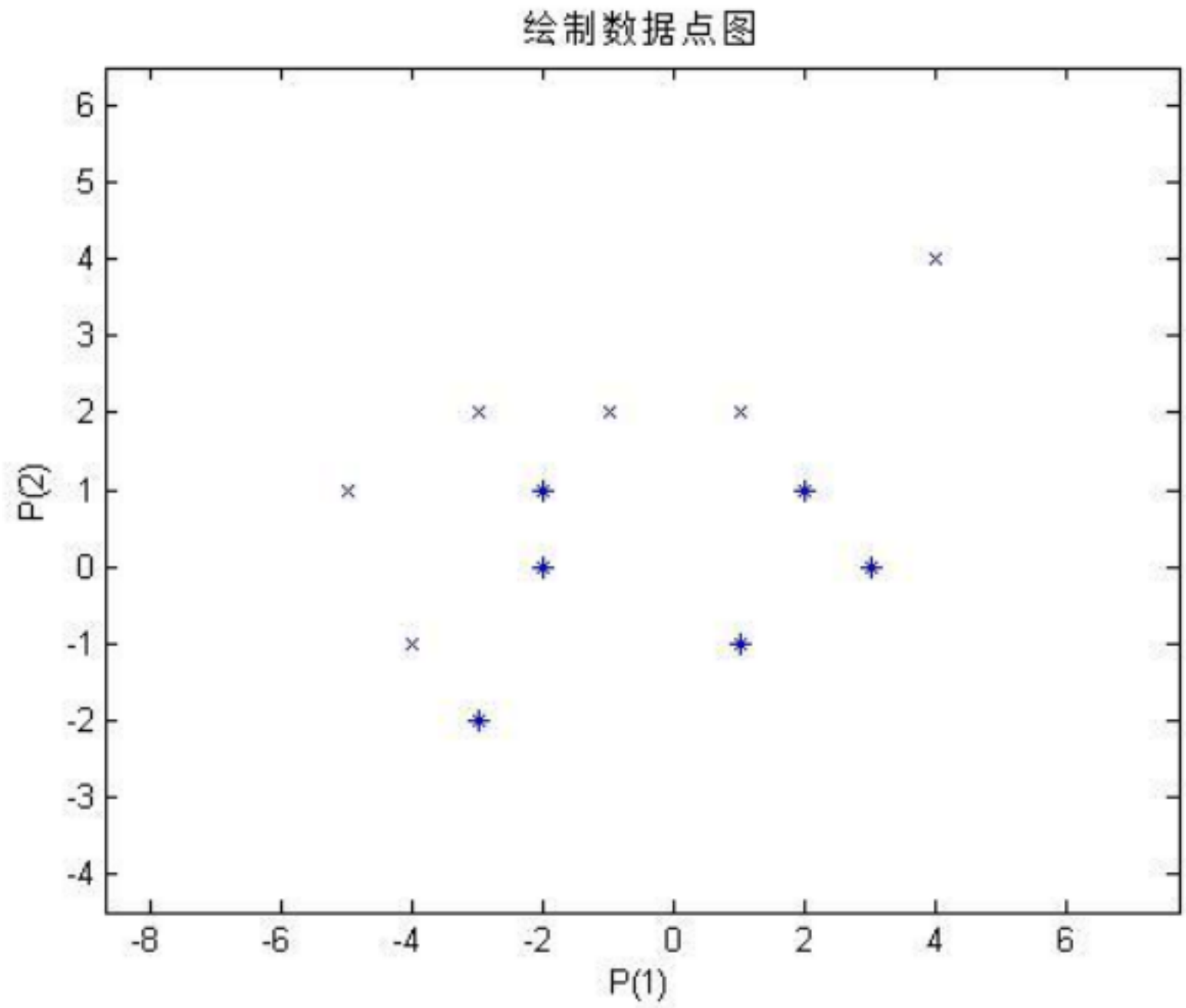
结果分析： 由上述结果可知， 奇异样本会严重影响感知器的训练速度， 增加迭代次数。使用“ 感知器归一化学习算法 ”即 learnpn 函数能有效消除奇异样本的影响。

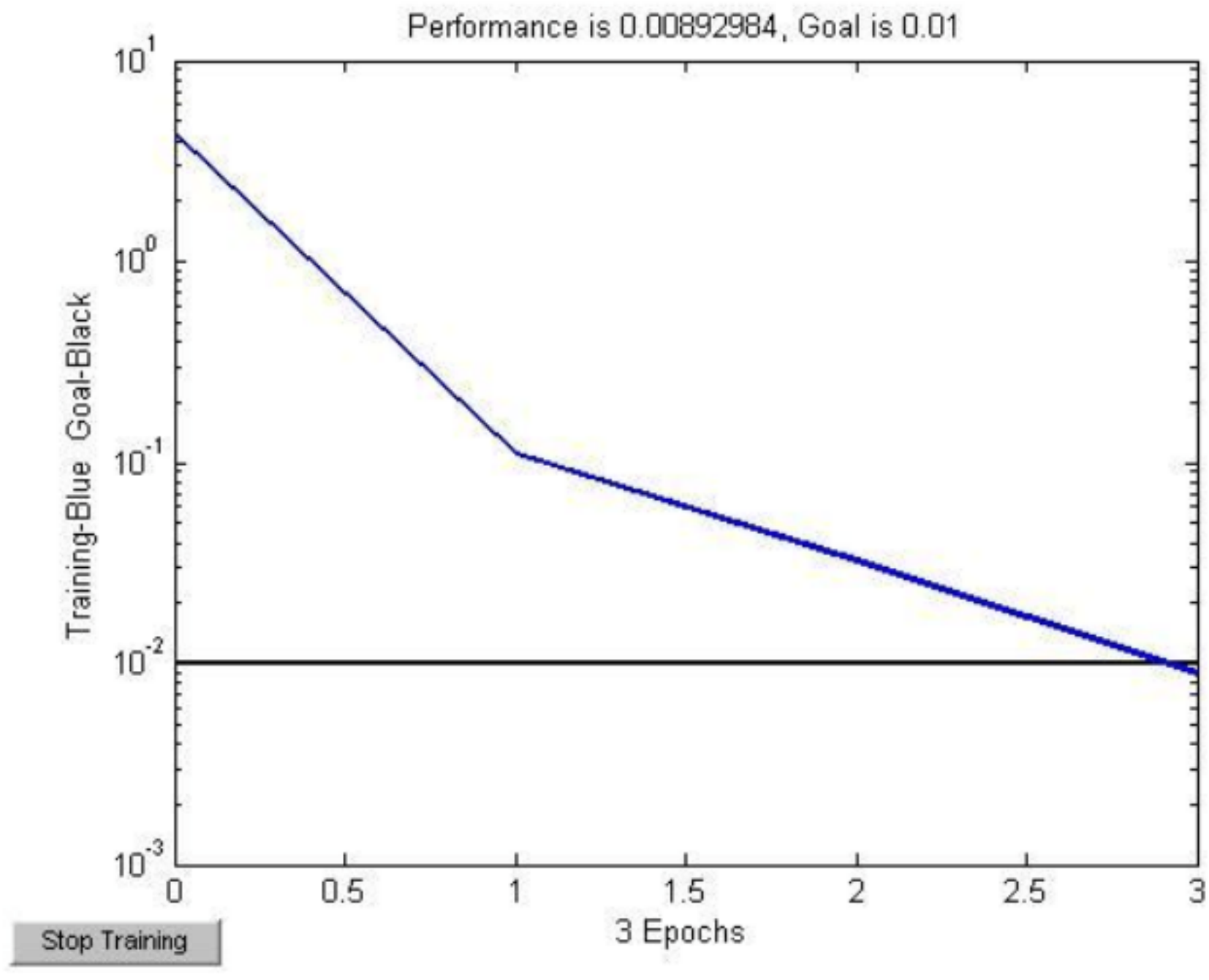
3、BP 网实验：利用 BP 网对上述线性不可分样本集进行分类

运行时间： Elapsed time is 1.417897 seconds.

```
A =
   Columns 1 through 9
    1.0415    0.9667    0.9916    0.9907    0.0319    0.6505    0.0481
-0.0433    0.0519
    1.0794    1.0450    1.0804    1.0389    1.0097    1.0512    1.0314
1.0329    1.0310
   Columns 10 through 12
   -0.0019    0.9765    0.2385
    0.9651    1.0132    1.0195
MSE =
    0.0089
```

实验截图如下：





由图可看出输入的样本矢量是线性不可分的，故利用 train 函数针对输入矢量 P 建立的相应的 BP训练网络。 BP 网络不像感知器能训练出稳定结果， 每次运行 BP 网络进行训练都可能得到不同的结果。程序中建立的网络包括输入层为三层，隐层节点数为 9，定义的训练误差为 0.01 ，最大迭代次数为 100，所得到的训练结果与目标的误差收敛于程序中定义的误差，故经过训练的 BP网络能够对线性不可分的网络进行分类。

参考教材

[1] 边肇祺，张学工．模式识别．清华大学出版社， 2000

[2] 董长虹． Matlab 神经网络与应用．国防工业出版社， 2005

[3] 冈萨雷斯．数字图像处理（第二版） ．电子工业出版社， 2005

[4] 徐飞，施晓红． MATLAB应用图像处理．西安电子科技大学出版社， 2002