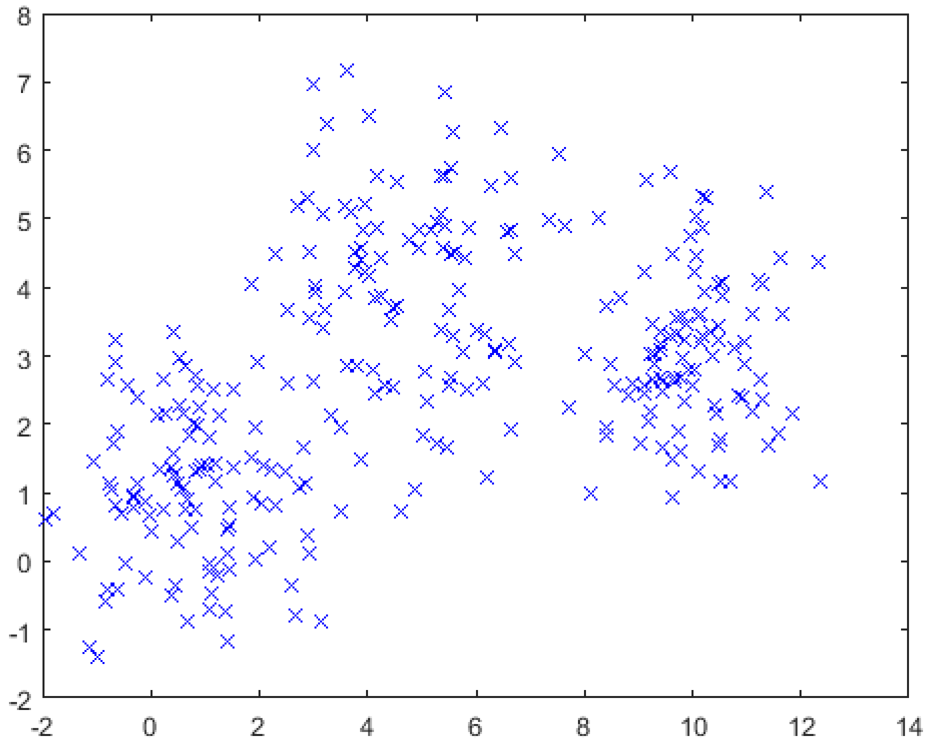
Contents

```
close all;
clear all;
clc;
```

产生样本点

```
N=300;
noise=0;
[data]=data_gen(N,noise);
% 对数据展示
plot(data(:,1),data(:,2),'bx')
```



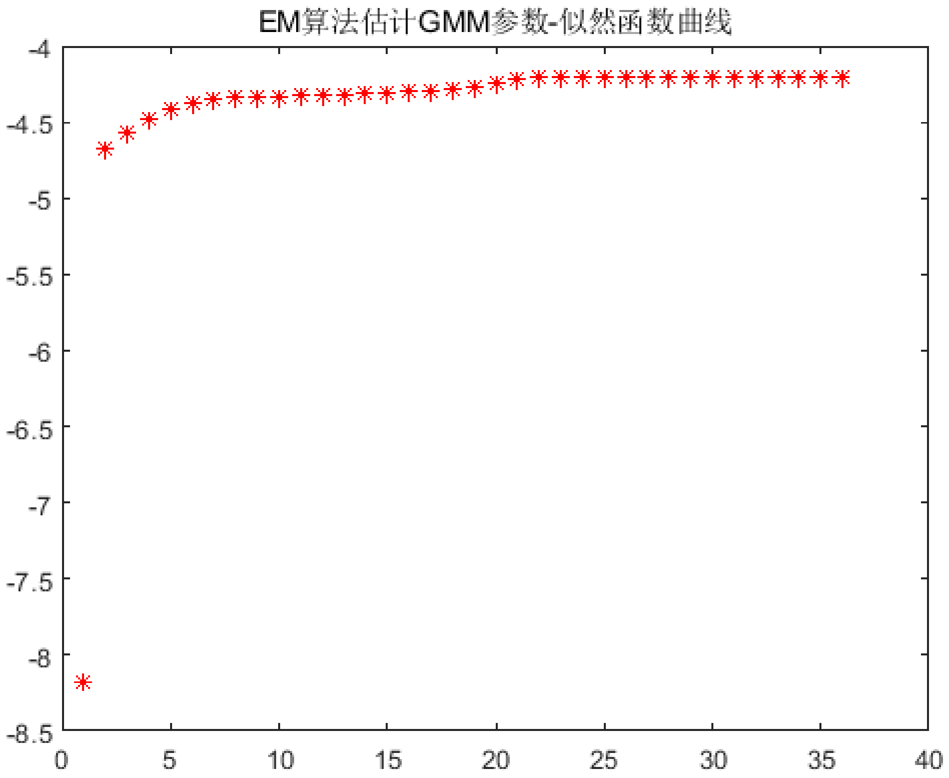
进行参数估计并比对结果

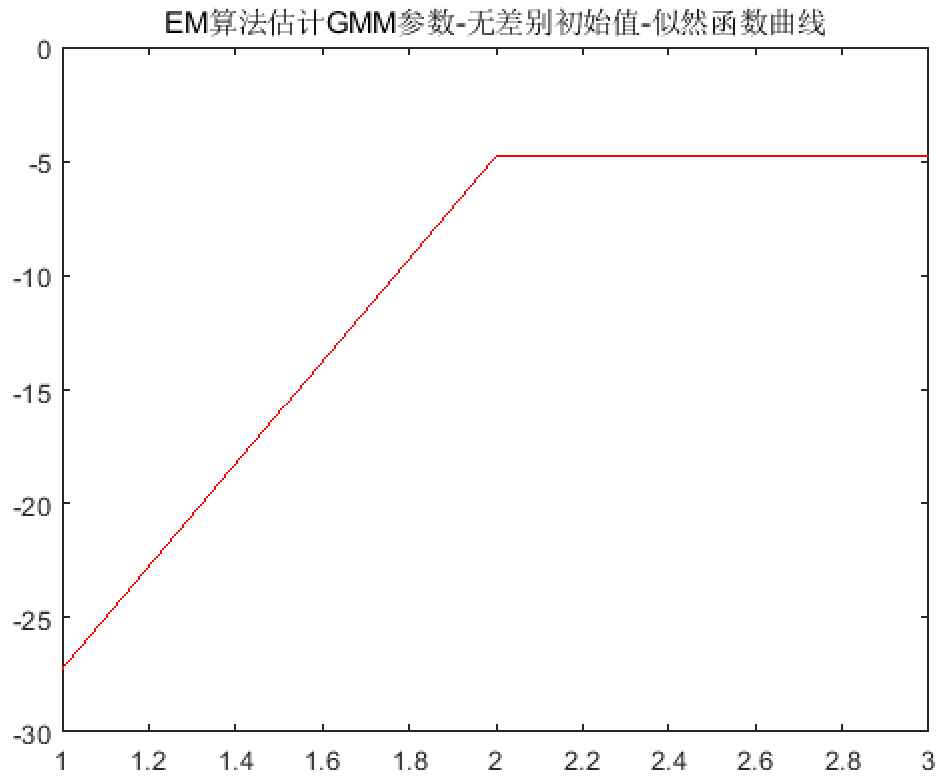
```
disp('--------------------原始参数值-----------------');
mu1 = [10,3]';%数学期望
mu2 = [1,1]';
mu3 = [5,4]';
sigma1=[1,0;0,1];%协方差矩阵
sigma2=[1.5,0;0,1.5];
```

```
sigma3=[2,0;0,2];
mu_ori=[mu1,mu2,mu3]
sigma1=[1,0;0,1];%协方差矩阵
sigma2=[3,0;0,3];
sigma3=[2,0;0,2];
sigma_ori=[sigma1,sigma2,sigma3]
w=[1,1,1]./3

disp('--------------------手动初始化------------------');
epson =1e-10;
% 手动初始化
mu1 = [3,5]';%数学期望
mu2 = [2,1]';
mu3 = [0,3]';
mu=[mu1,mu2,mu3]
sigma1=[1,0;0,1];%协方差矩阵
sigma2=[3,0;0,3];
sigma3=[2,0;0,2];
sigma=[sigma1,sigma2,sigma3]
phi=[0.5,0.4,0.1]
disp('---------------------估计值------------------')
[L,mu_1,sigma_1,weight_1]=EM_GMM(data,mu,sigma,phi,epson);

mu_1
sigma_1
weight_1
figure
plot(L,'R*')
title('EM算法估计GMM参数-似然函数曲线');




disp('---------------------完全无差别的初始值设置------------------')
mu1 = [0,0]';%数学期望
mu2 = [0,0]';
mu3 = [0,0]';
mu=[mu1,mu2,mu3]
sigma1=[1,0;0,1];%协方差矩阵
sigma2=[1,0;0,1];
sigma3=[1,0;0,1];
sigma=[sigma1,sigma2,sigma3]
phi=[1/3,1/3,1/3]


disp('---------------------无差别初始值时的估计值------------------')
[L,mu_1,sigma_1,weight_1]=EM_GMM(data,mu,sigma,phi,epson);

mu_1
sigma_1
weight_1
figure
plot(L,'R')
title('EM算法估计GMM参数-无差别初始值-似然函数曲线');
```

```
--------------------原始参数值------------------

mu_ori =

    10     1     5
     3     1     4
```

```
sigma_ori =

     1     0     3     0     2     0
     0     1     0     3     0     2


w =

    0.3333    0.3333    0.3333

--------------------手动初始化------------------

mu =

     3     2     0
     5     1     3


sigma =

     1     0     3     0     2     0
     0     1     0     3     0     2


phi =

    0.5000    0.4000    0.1000

---------------------估计值------------------
```



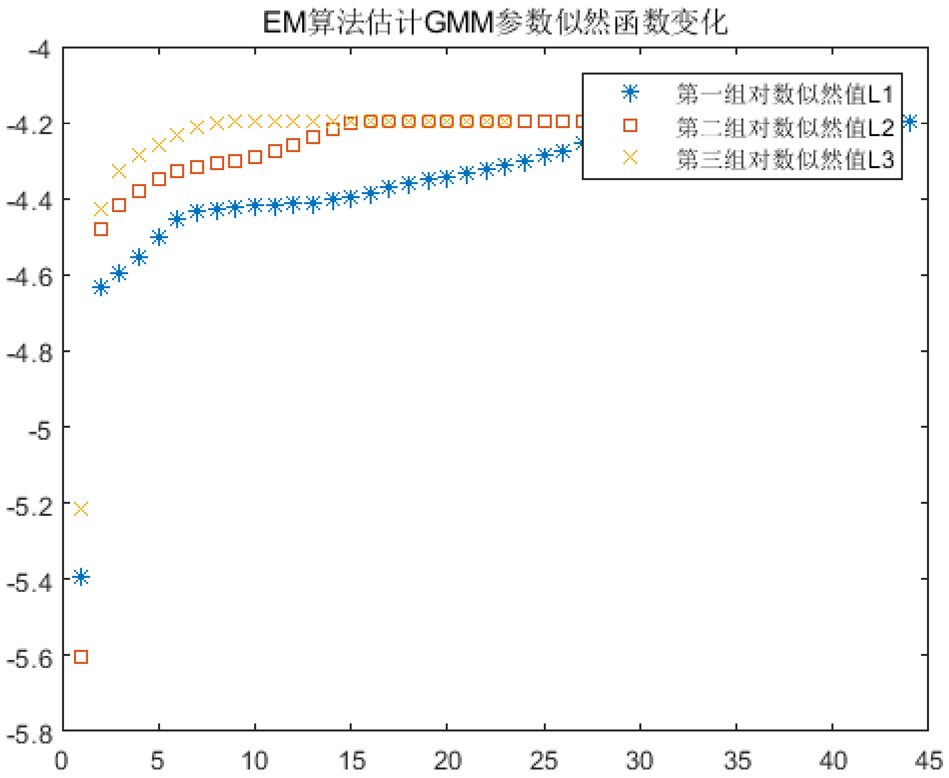


随机初始化参数并比对结果

```
% 第一组
[mu,sigma,weight]=param_gen();
[L1,mu_1,sigma_1,weight_1]=EM_GMM(data,mu,sigma,phi,epson);
mu_1
sigma_1
weight_1
% 第二组
```

```
[mu,sigma,weight]=param_gen()
[L2,mu_2,sigma_2,weight_2]=EM_GMM(data,mu,sigma,phi,epson);
mu_2
sigma_2
weight_2
% 第三组
[mu,sigma,weight]=param_gen()
[L3,mu_3,sigma_3,weight_3]=EM_GMM(data,mu,sigma,phi,epson);
mu_3
sigma_3
weight_3

% 对比不同结果
figure
plot(L1','*');
hold on
plot(L2','s');
plot(L3','x');

legend('第一组对数似然值L1','第二组对数似然值L2','第三组对数似然值L3');

title('EM算法估计GMM参数似然函数变化');
```

```
mu_1 =

    9.9771    0.7502    4.7702
    3.0438    1.0296    4.0296


sigma_1 =

    0.8790         0    1.3397         0    1.8619         0
         0    1.1983         0    1.2197         0    2.0419


weight_1 =

    0.3167    0.3512    0.3322


mu_2 =

    4.7702    9.9771    0.7502
    4.0296    3.0438    1.0296


sigma_2 =

    1.8619         0    0.8790         0    1.3397         0
         0    2.0419         0    1.1983         0    1.2197


weight_2 =

    0.3322    0.3167    0.3512


mu_3 =

    0.7502    9.9771    4.7702
    1.0296    3.0438    4.0296


sigma_3 =
    1.3397         0    0.8790         0    1.8619         0
         0    1.2197         0    1.1983         0    2.0419


weight_3 =

    0.3512    0.3167    0.3322
```

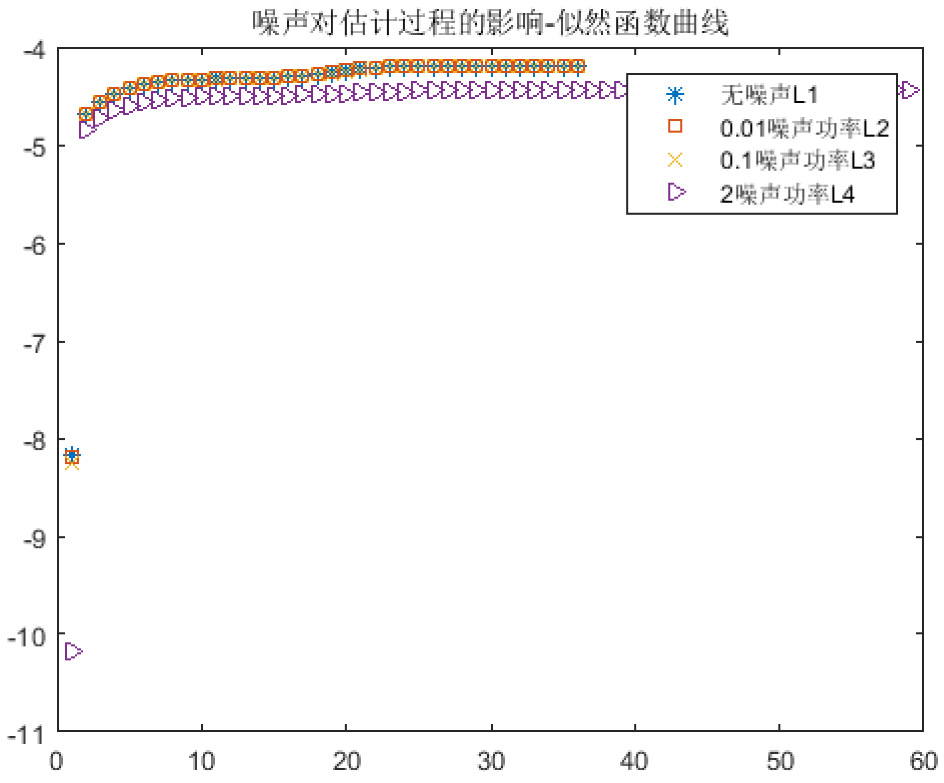


对数据增加噪声并分析结果

```
disp('---------------------噪声对估计的影响------------------')
% 设置初始参数值
mu1 = [3,5]';%数学期望
mu2 = [2,1]';
mu3 = [0,3]';
mu=[mu1,mu2,mu3];
sigma1=[1,0;0,1];%协方差矩阵
sigma2=[3,0;0,3];
sigma3=[2,0;0,2];
sigma=[sigma1,sigma2,sigma3];
phi=[0.5,0.4,0.1];
% 原始数据组
data;
[L1,~,~,~]=EM_GMM(data,mu,sigma,phi,epson);

% 加噪声第一组
[m,n]=size(data);
noise=0.01;
ndata=noise*rand(m,n);
data=data+ndata;
[L2,~,~,~]=EM_GMM(data,mu,sigma,phi,epson);
% 加噪声第二组
 noise=0.1;
ndata=noise*rand(m,n);
```

```
data=data+ndata;
[L3,~,~,~]=EM_GMM(data,mu,sigma,phi,epson);

% 加噪声第二组
 noise=2;
ndata=noise*rand(m,n);
data=data+ndata;
[L4,~,~,~]=EM_GMM(data,mu,sigma,phi,epson);




figure
plot(L1','*');
hold on
plot(L2','s');
plot(L3','x');
plot(L4','>');

legend('无噪声L1','0.01噪声功率L2','0.1噪声功率L3','2噪声功率L4');

title('噪声对估计过程的影响-似然函数曲线');
```

---------------------噪声对估计的影响------------------



生成数据函数

```
function   [data]=data_gen(N,noise)
mu1 = [10,3]';%数学期望
mu2 = [1,1]';
mu3 = [5,4]';
sigma1=[1,0;0,1];%协方差矩阵
sigma2=[1.5,0;0,1.5];
sigma3=[2,0;0,2];
w=[1,1,1]./3;
```

```
r1=mvnrnd(mu1,sigma1,N);
r2=mvnrnd(mu2,sigma2,N);
r3=mvnrnd(mu3,sigma3,N);
rawdata=[r1,r2,r3];
% 产生随机数
rate = floor(rand(N,1)*3)+1;
data=zeros(N,2);
% 取数据
    for i=1:3
        idx=find(rate==i);
        data(idx,:)=rawdata(idx,i*2-1:i*2);
    end
% 增加噪声
[m,n]=size(data);
ndata=noise*rand(m,n);
data=data+ndata;

end
```

随机初始化参数

```
function   [mu,sigma,weight]=param_gen()
% 产生均值
mu=floor(rand(2,3)*10);
%协方差矩阵
s1=ceil(rand(1)*10)*eye(2);
s2=ceil(rand(1)*10)*eye(2);
s3=ceil(rand(1)*10)*eye(2);
sigma=[s1,s2,s3];
% 权值
w_i=rand(1,3);
weight=w_i/sum(w_i);
end
```

```
mu =

     2     6     3
     8     6     0


sigma =

    10     0     7     0     3     0
     0    10     0     7     0     3


weight =

    0.0747    0.1950    0.7304




mu =

     2     9     4
     4     1     6


sigma =

     6     0     7     0     3     0
     0     6     0     7     0     3


weight =

    0.1966    0.3708    0.4325
```

使用EM算法对GMM参数进行估计

```
function   [L,mu_s,sigma_s,weight_s]=EM_GMM(data,mu,sigma,phi,epson)
N=length(data);
T=5000;
w = zeros(N,3);
muarr=[];
sigmarr=[];
phiarr=[];
error=100;
L(1)=1;
pos=2;

% while(true)
for y=1:1000
    % Expectation
    for k = 1 : 3
        w(:,k) = phi(k)*mvnpdf(data,mu(:,k)',sigma(:,k*2-1:k*2));%  对于每一个样本，对参数w进行估计
    end
     % 中间穿插计算似然函数
    L(pos)=sum(log(sum(w,2)))/N;
    err=L(pos)-L(pos-1);
    if(abs(err)<epson)
        break;
    end
    pos=pos+1;

    w = w./repmat(sum(w,2),[1 3]);

    % Maximization
    for k = 1 : 3
        mu(:,k) = (w(:,k)'*data / sum(w(:,k)))';
%         sigma(:,k*2-1:k*2) = w(:,k)'*((data-mu(:,k)')'*(data-mu(:,k)')) / sum(w(:,k));
%         temp= kron(w(:,k),((data-mu(:,k)')'*(data-mu(:,k)'))) / sum(w(:,k));% 对方差求解
        % 有一个矩阵求和过程
%         sigma(:,k*2-1:k*2)= sqrt(reshape(sum(reshape(temp',4,N),2),2,2));

%  直接构造矩阵
        cov_mat=diag(w(:,k)'*((data-mu(:,k)').*(data-mu(:,k)')))/sum(w(:,k));
        sigma(:,k*2-1:k*2)=cov_mat;
        phi(k) = sum(w(:,k)) / N;
    end
    muarr = [muarr;reshape(mu,1,6)];
    sigmarr= [sigmarr;reshape(sigma,1,12)];
    phiarr= [phiarr;phi];
    y=y+1;
end

mu_s=mu;
sigma_s=sigma;
weight_s=phi;
L=L(2:end);
end
```

```
mu_1 =

    4.7702    9.9771    0.7502
    4.0296    3.0438    1.0296


sigma_1 =

    1.8619         0    0.8790         0    1.3397         0
         0    2.0419         0    1.1983         0    1.2197


weight_1 =

    0.3322    0.3167    0.3512

---------------------完全无差别的初始值设置------------------

mu =

     0     0     0
     0     0     0


sigma =

     1     0     1     0     1     0
     0     1     0     1     0     1


phi =

    0.3333    0.3333    0.3333

---------------------无差别初始值时的估计值------------------

mu_1 =

    5.0073    5.0073    5.0073
    2.6639    2.6639    2.6639


sigma_1 =

   15.5717         0   15.5717         0   15.5717         0
         0    3.0892         0    3.0892         0    3.0892


weight_1 =

    0.3333    0.3333    0.3333
```

Published with MATLAB® R2017b