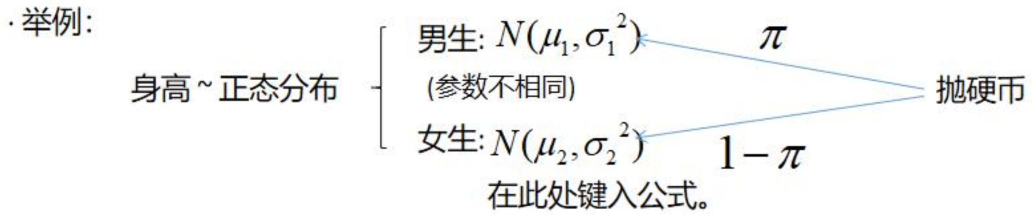
Table of Contents

1 EM 算法(Expectation Maximization)引入

核心思想：
含有**隐变量（缺失数据）**的概率模型的**极大似然估计**或**极大后验概率估计法**



已知测得 100 个人身高，在测身高之前先抛硬币：如果是正面则测男生，如果是反面则测女生。然后用统计方法估计参数 $\pi,\mu_1,\sigma_1^{\ 2},\mu_2,\sigma_2^{\ 2}$ 的极大似然估计?

即 已知身高 x1,x2,...,x100,求下面分布函数参数的MLE

$$X\sim\pi\phi(\frac{x-\mu_1}{\sigma_1})+(1-\pi)\phi(\frac{x-\mu_2}{\sigma_2})$$

缺失数据：性别

$$(x_1, x_2, \ldots, x_n) \overset{iid}{\sim} X$$

极大似然估计（MLE）

$1.L(x;\theta) = \prod_{i=1}^{n} P(x_i;\theta)$

$2.\frac{\partial L(x;\theta)}{\partial \theta} = 0$

$3.\hat{\theta}_{MLE}$是上述方程的解

常取 对数 log 来简化

EM算法：
引入隐变量
（引入**隐变量的期望**）

解决方法：换成乘法

在上题中的表现

$$1.P(x) = \boxed{\pi \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} + (1-\pi)\frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}}}$$

难处理

$$2.\log(x_i;\theta) = \log \prod_{i=1}^{n} P(x_i;\theta) = \sum_{i=1}^{n} \log(\_\_\_\_ + \_\_\_\_)$$

$$3.\frac{\partial \log L(x;\theta)}{\partial \theta} = 0$$

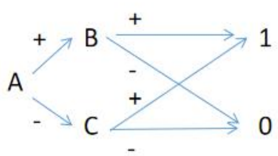
超定方程
（有无穷多解；无唯一解析解）

2 李航书本 EM 算法引入

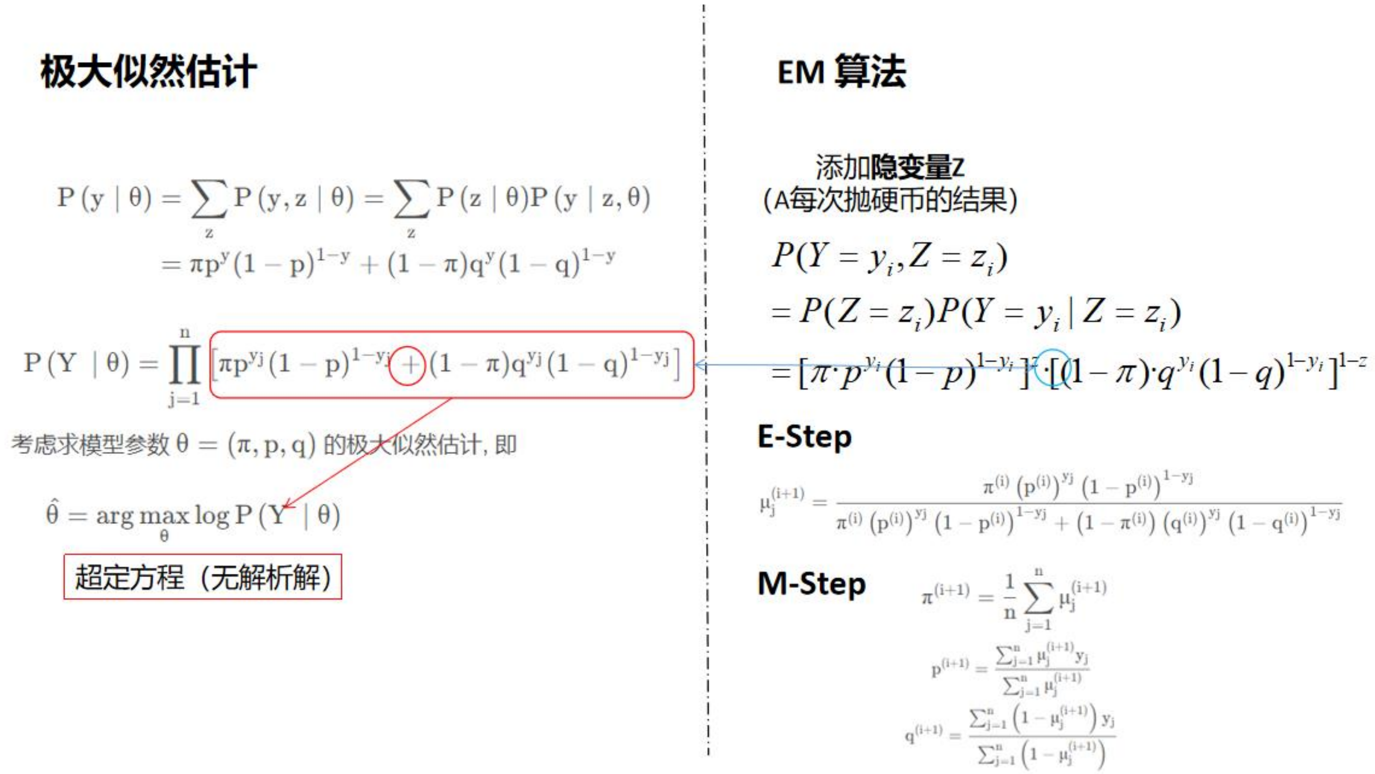
三硬币模型 假设有 3 枚硬币，分别记作 A, B, C 。这些硬币正面出现 的概率分别是 π, p 和 q。进行如下掷硬币试验: 先掷硬币 A，根据其结果选出硬币 B 或硬币 C，正面选硬币 B，反面选硬币 C；然后掷选出的硬币，掷硬币的结果，出现正 面记作 1，出现反面记作 0；独立地重复 n 次试验 (这里，$n=10$)，观测结果如下:

$$1,1,0,1,0,0,1,0,1,1$$

假设只能观测到掷硬币的结果，不能观测掷硬币的过程。问如何估计三硬币正面出现的概率，即三硬币模型的参数。



$$P(A=\text{正}) = \pi$$
$$P(B=\text{正}) = p$$
$$P(C=\text{正}) = q$$



3 EM 算法

==

输入: 观测变量数据 Y, 隐变量数据 Z, 联合分布 $P(Y, Z \mid \theta)$, 条件分布 $P(Z \mid Y, \theta)$;

输出: 模型参数 θ 。

1. 选择参数的初值 $\theta^{(0)}$ ，开始迭代;
2. E 步: 记 $\theta^{(\mathrm{i})}$ 为第 i 次迭代参数 θ 的估计值, 在第 $i+1$ 次迭代的 E 步, 计算
$$\begin{aligned} \mathrm{Q}\left(\theta, \theta^{(\mathrm{i})}\right) &= \mathrm{E}_{\mathrm{Z}}\left[\log \mathrm{P}(\mathrm{Y}, \mathrm{Z} \mid \theta) \mid \mathrm{Y}, \theta^{(\mathrm{i})}\right] \\ &= \sum_{\mathrm{Z}} \log \mathrm{P}(\mathrm{Y}, \mathrm{Z} \mid \theta) \mathrm{P}\left(\mathrm{Z} \mid \mathrm{Y}, \theta^{(\mathrm{i})}\right) \end{aligned}$$
这里, $\mathrm{P}\left(\mathrm{Z} \mid \mathrm{Y}, \theta^{(\mathrm{i})}\right)$ 是在给定观测数据 Y 和当前的参数估计 $\theta^{(\mathrm{i})}$ 下隐变量数据 Z 的条概率分布;
3. M 步: 求使 $\mathrm{Q}\left(\theta, \theta^{(\mathrm{i})}\right)$ 极大化的 θ, 确定第 $\mathrm{i}+1$ 次迭代的参数的估计值 $\theta^{(\mathrm{i}+1)}$
$$\theta^{(\mathrm{i}+1)}=\arg \max_\theta \mathrm{Q}\left(\theta, \theta^{(\mathrm{i})}\right)$$
4. 重复第 (2)步和第 (3) 步, 直到收敛。

==

4 抛硬币例子实现

9.1、如例9.1的三硬币模型。假设观测数据不变，试选择不同的初值，例如$\pi^{(0)}=0.46, p^{(0)}=0.55, q^{(0)}=0.67$，求模型参数$\theta=(\pi, p, q)$的极大似然估计。

- ## 4.1 李航9.1手算求解一步

- **4.1.1 E 步**

$$\mu_1^{(1)}=\frac{\pi^{(0)}\left(p^{(0)}\right)^{y_1}\left(1-p^{(0)}\right)^{1-y_1}}{\pi^{(0)}\left(p^{(0)}\right)^{y_1}\left(1-p^{(0)}\right)^{1-y_1}+\left(1-\pi^{(0)}\right)\left(q^{(0)}\right)^{y_1}\left(1-q^{(0)}\right)^{1-y_1}}=\frac{0.46\cdot 0.55}{0.46\cdot 0.55+(1-0.46)\cdot 0.}$$

同理，其他结果为1的结果同上，即$\mu_2^{(1)},\mu_4^{(1)}$，$\mu_7^{(1)}$，$\mu_9^{(1)}$，$\mu_{10}^{(1)}$值和$\mu_1^{(1)}$相同

$$\mu_3^{(1)}=\frac{\pi^{(0)}\left(p^{(0)}\right)^{y_3}\left(1-p^{(0)}\right)^{1-y_3}}{\pi^{(0)}\left(p^{(0)}\right)^{y_3}\left(1-p^{(0)}\right)^{1-y_3}+\left(1-\pi^{(0)}\right)\left(q^{(0)}\right)^{y_3}\left(1-q^{(0)}\right)^{1-y_3}}=\frac{0.46\cdot(1-0.55)}{0.46\cdot(1-0.55)+(1-0.46)\cdot(}$$

同理，其他结果为0的结果同上，即$\mu_5^{(1)},\mu_6^{(1)}$，$\mu_8^{(1)}$值和$\mu_3^{(1)}$相同

所以

$$\hat{\mu}_1=[0.412,0.412,0.537,0.412,0.537,0.537,0.412,0.537,0.412,0.412]$$

- **4.1.2 M 步**

$$\pi^{(1)}=\frac{1}{n}\sum_{j=1}^{n}\mu_j^{(1)}=\frac{6\cdot 0.412+4\cdot 0.537}{10}=0.462$$

$$p^{(1)}=\frac{\sum_{j=1}^{n}\mu_j^{(1)}y_j}{\sum_{j=1}^{n}\mu_j^{(1)}}=\frac{6\cdot 0.412\cdot 1+4\cdot 0.537\cdot 0}{6\cdot 0.412+4\cdot 0.537}=0.535$$

$$q^{(1)}=\frac{\sum_{j=1}^{n}\left(1-\mu_j^{(1)}\right)y_j}{\sum_{j=1}^{n}\left(1-\mu_j^{(1)}\right)}=\frac{6\cdot(1-0.412)\cdot 1+4\cdot(1-0.537)\cdot 0}{6\cdot(1-0.412)+4\cdot(1-0.537)}=0.656$$

故第一次迭代结果是:

$$\pi^{(1)}=0.462$$
$$p^{(1)}=0.535$$
$$q^{(1)}=0.656$$

- **4.2 python代码实现上例**

In [16]:

```python
import numpy as np
#import pandas as pd


def dist(a, b):
    return np.max(np.abs(a - b))


class EM_algorithm(object):
    def __init__(self, Y, pi, p, q, error=1e-04):
        self.y = Y
        self.pi = pi
        self.p = p
        self.q = q
        self.epslon = error

    def em_esti(self):
        result = list()
        number = 0
        flag = True
        #old=[self.pi,self.p,self.q]
        #new=np.zeros(shape=3)
        pi = self.pi
        p = self.p
        q = self.q
        while flag:
            #E-step:
            print("mu值是")
            mu = np.zeros(shape=len(self.y))
            for i in range(len(self.y)):
                temp1 = pi * p**self.y[i] * (1 - p)**(1 - self.y[i])
                temp2 = (1 - pi) * q**self.y[i] * (1 - q)**(1 - self.y[i])
                mu[i] = temp1 / (temp1 + temp2)
                print(mu[i])
            #M-step:
            ppi = np.sum(mu) / len(self.y)
            pp = np.dot(mu, self.y) / np.sum(mu)
            qq = np.dot(1 - mu, self.y) / np.sum(1 - mu)
            print('===============')

            number += 1
            result.append(np.array([number, ppi, pp, qq]))
            flag = (dist(np.array([pi, p, q]), np.array([ppi, pp, qq])) >
                    self.epslon)
            pi = ppi
            p = pp
            q = qq
            print("迭代结果是")
            print(pi)
            print(p)
            print(q)
            print('===============')
        return result


Y_train = np.array([1, 1, 0, 1, 0, 0, 1, 0, 1, 1])
#theta0=np.array([0.4,0.6,0.7])

eme = EM_algorithm(Y_train, 0.46, 0.55, 0.67, 0.00001)
```

```
print(eme.em_esti())
```

```
mu值是
0.41151594014313597
0.41151594014313597
0.5373831775700935
0.41151594014313597
0.5373831775700935
0.5373831775700935
0.41151594014313597
0.5373831775700935
0.41151594014313597
0.41151594014313597
===============
迭代结果是
0.461862835113919
0.5345950037850113
0.6561346417857324
===============
mu值是
0.411515940143136
0.411515940143136
0.5373831775700935
0.411515940143136
0.5373831775700935
0.5373831775700935
0.411515940143136
0.5373831775700935
0.411515940143136
0.411515940143136
===============
迭代结果是
0.461862835113919
0.5345950037850113
0.6561346417857324
===============
[array([1.        , 0.46186284, 0.534595  , 0.65613464]), array([2.        , 0.46186
284, 0.534595  , 0.65613464])]
```

5 混合高斯分布的求解

$$X \sim \pi\phi(\frac{x-\mu_1}{\sigma_1}) + (1-\pi)\phi(\frac{x-\mu_2}{\sigma_2})$$

1. E 步:

$$E(z_i \mid \theta^{(t)}) = \eta_i^{(t+1)} = \frac{\pi^{(t)}\phi(\frac{y_i-\mu_1^{(t)}}{\sigma_1^{(t)}})}{\pi^{(t)}\phi(\frac{y_i-\mu_1^{(t)}}{\sigma_1^{(t)}}) + (1-\pi^{(t)})\phi(\frac{y_i-\mu_2^{(t)}}{\sigma_2^{(t)}})}$$

2. M 步:

$$\pi^{(t+1)} = \frac{1}{n}\sum_{i=1}^{n}\eta_i^{(t+1)}$$

$$\mu_1^{(t+1)} = \frac{\sum_{i=1}^{n}\eta_i^{(t+1)}y_i}{\sum_{i=1}^{n}\eta_i^{(t+1)}} \qquad \mu_2^{(t+1)} = \frac{\sum_{i=1}^{n}(1-\eta_i^{(t+1)})y_i}{\sum_{i=1}^{n}(1-\eta_i^{(t+1)})}$$

$$\sigma_1^{2(t+1)} = \frac{\sum_{i=1}^{n}\eta_i^{(t+1)}(y_i-\mu_1^{(t+1)})^2}{\sum_{i=1}^{n}\eta_i^{(t+1)}} \qquad \sigma_2^{2(t+1)} = \frac{\sum_{i=1}^{n}(1-\eta_i^{(t+1)})(y_i-\mu_2^{(t+1)})^2}{\sum_{i=1}^{n}(1-\eta_i^{(t+1)})}$$

- ## 5.1 李航书本9.3手算求解一步

9.3、已知观测数据-67，-48，6，8，14，16，23，24，28，29，41，49，56，60，75，试估计两个分量的高斯混合模型的5个参数。

- ### 5.1.1 取初值:

为了更快收敛，假设初值来自两个不同的正态分布总体，前8个数据为一组，后7个为一组，分别计算出其对应的样本均值和方差。

$$\pi^{(0)} = \frac{8}{15} = 0.533$$

$$\mu_1^0 = \frac{-67-48+6+8+14+16+23+24}{8} = -3$$

$$\sigma_1^{2(0)} = \frac{(-67+3)^2+(-48+3)^2+(6+3)^2+(8+3)^2+(14+3)^2+(16+3)^2+(23+3)^2+(24 \cdot}{8}$$

$$\mu_2^0 = \frac{28+29+41+49+56+60+75}{7} = 48.286$$

$$\sigma_2^{2(0)} = \frac{(28-48.486)^2+(29-48.486)^2+(41-48.486)^2+(49-48.486)^2+(56-48.486)^2+(6}{7}$$

- ### 5.1.2 E 步:

$$\eta_1^{(1)} = \frac{\pi^{(0)}\phi(\frac{y_i-\mu_1^{(0)}}{\sigma_1^{(0)}})}{\pi^{(0)}\phi(\frac{y_i-\mu_1^{(0)}}{\sigma_1^{(0)}}) + (1-\pi^{(0)})\phi(\frac{y_i-\mu_2^{(0)}}{\sigma_2^{(0)}})} = 0.232$$

同理，剩下的 η 为

$$\eta_2 = 0.227, \eta_3 = 0.217, \eta_4 = 0.216$$
$$\eta_5 = 0.216, \eta_6 = 0.215, \eta_7 = 0.215$$
$$\eta_8 = 0.215, \eta_9 = 0.215, \eta_{10} = 0.215$$
$$\eta_{11} = 0.214, \eta_{12} = 0.214, \eta_{13} = 0.214$$
$$\eta_{14} = 0.214, \eta_{15} = 0.215$$

- **5.1.3 M 步：**

$$\pi^{(1)} = \frac{\sum_{i=1}^{n} \eta_i}{n}$$
$$= \frac{0.232 + 0.227 + 0.217 + 0.216 + 0.216 + 0.215 + 0.215 + 0.215 + 0.215 + 0.215 + 0.214 + 0.214 +}{15}$$

$$\mu_1^{(1)} = \frac{\sum_{i=1}^{n} \eta_i y_i}{\sum_{i=1}^{n} \eta_i} =$$
$$= \frac{-67 \cdot 0.232 - 48 \cdot 0.227 + 6 \cdot 0.217 + 8 \cdot 0.216 + 14 \cdot 0.216 + 16 \cdot 0.215 + 23 \cdot 0.215 + 24 \cdot 0.215 + 28 \cdot 0.215 + 29 \cdot 0.215 + 41 \cdot 0.214 + 49 \cdot 0.214 + 56 \cdot 0.214 + 60 \cdot 0.2}{0.232 + 0.227 + 0.217 + 0.216 + 0.216 + 0.215 + 0.215 + 0.215 + 0.215 + 0.215 + 0.214 + 0.214 + 0.214 + 0.214 + 0.215}$$

$$\mu_2^{(1)} = \frac{\sum_{i=1}^{n} (1 - \eta_i) y_i}{\sum_{i=1}^{n} (1 - \eta_i)}$$
$$= \frac{-67 \cdot 0.768 - 48 \cdot 0.773 + 6 \cdot 0.783 + 8 \cdot 0.784 + 14 \cdot 0.784 + 16 \cdot 0.785 + 23 \cdot 0.785 + 24 \cdot 0.785 + 28 \cdot 0.785 + 29 \cdot 0.785 + 41 \cdot 0.786 + 49 \cdot 0.786 + 56 \cdot 0.786 + 60 \cdot 0.7}{0.768 + 0.773 + 0.783 + 0.784 + 0.784 + 0.785 + 0.785 + 0.785 + 0.785 + 0.785 + 0.786 + 0.786 + 0.786 + 0.786 + 0.785}$$

$$\sigma_1^{2(1)} = \frac{\sum_{i=1}^{n} \eta_i (y_i - \mu_1^{(1)})^2}{\sum_{i=1}^{n} \eta_i}$$
$$= \frac{\begin{array}{l}(-67 - 20.156)^2 \cdot 0.232 - (48 - 20.156)^2 \cdot 0.227 + (6 - 20.156)^2 \cdot 0.217 + (8 - 20.156)^2 \cdot 0.216 + (14 - 20.156)^2 \cdot 0.216 + (16 - 20.156)^2 \cdot 0.215 + (23 - 20.156 \\ + (29 - 20.156)^2 \cdot 0.215 + (41 - 20.156)^2 \cdot 0.214 + (49 - 20.156)^2 \cdot 0.214 + (56 - 20.156)^2 \cdot 0.214 + (60 - 20.156)^2 \cdot 0.214 + (75 - 20.156)^2 \cdot 0.215\end{array}}{0.232 + 0.227 + 0.217 + 0.216 + 0.216 + 0.215 + 0.215 + 0.215 + 0.215 + 0.215 + 0.214 + 0.214 + 0.214 + 0.21}$$
$$= 1374.615$$

$$\sigma_2^{2(1)} = \frac{\sum_{i=1}^{n} (1 - \eta_i)(y_i - \mu_2^{(1)})^2}{\sum_{i=1}^{n} (1 - \eta_i)}$$
$$= \frac{\begin{array}{l}(-67 - 21.149)^2 \cdot 0.768 - (48 - 21.149)^2 \cdot 0.773 + (6 - 21.149)^2 \cdot 0.783 + (8 - 21.149)^2 \cdot 0.784 + (14 - 21.149)^2 \cdot 0.784 + (16 - 21.149)^2 \cdot 0.785 + (23 - 21.149 \\ + (29 - 21.149)^2 \cdot 0.785 + (41 - 21.149)^2 \cdot 0.786 + (49 - 21.149)^2 \cdot 0.786 + (56 - 21.149)^2 \cdot 0.786 + (60 - 21.149)^2 \cdot 0.786 + (75 - 21.149)^2 \cdot 0.785\end{array}}{0.768 + 0.773 + 0.783 + 0.784 + 0.784 + 0.785 + 0.785 + 0.785 + 0.785 + 0.785 + 0.786 + 0.786 + 0.786 + 0.78}$$
$$= 1317.004$$

即第一次迭代结果是

$$\pi^{(1)} = 0.217$$
$$\mu_1^{(1)} = 20.156$$
$$\mu_2^{(1)} = 21.149$$
$$\sigma_1^{2(1)} = 1374.615$$
$$\sigma_2^{2(1)} = 1317.004$$

- ## 5.2 python代码实现上例

In [24]:

```
import numpy as np

arr1 = [-67, -48, 6, 8, 14, 16, 23, 24]
arr2 = [28, 29, 41, 49, 56, 60, 75]
#求方差
arr_1 = np.mean(arr1)
arr_2 = np.mean(arr2)
#求方差
arr_var1 = np.var(arr1)
arr_var2 = np.var(arr2)
print("均值1为：%f" % arr_1)
print("方差1为：%f" % arr_var1)
print("均值2为：%f" % arr_2)
print("方差2为：%f" % arr_var2)
```

```
均值1为：-3.000000
方差1为：1047.250000
均值2为：48.285714
方差2为：249.632653
```

In [4]:

```python
import numpy as np
#import pandas as pd
from scipy import stats
#prob = stats.norm.pdf(x, mu, sigma)


def dist(a, b):
    return np.sqrt(np.dot(a - b, a - b))


class EM_MG(object):
    def __init__(self, Y, theta, mu1, sigma1, mu2, sigma2, error=1e-04):
        self.y = Y
        self.theta = theta
        self.mu1 = mu1
        self.sigma1 = sigma1
        self.mu2 = mu2
        self.sigma2 = sigma2
        self.epslon = error

    def em_estimate(self):
        result = list()
        number = 0
        flag = True
        #old=[self.pi,self.p,self.q]
        #new=np.zeros(shape=3)
        theta = self.theta
        mu1 = self.mu1
        sigma1 = self.sigma1
        mu2 = self.mu2
        sigma2 = self.sigma2
        while flag:
            #E-step:
            print('eta值为：')
            the = np.zeros(shape=len(self.y))
            for i in range(len(self.y)):
                temp1 = theta * stats.norm.pdf(self.y[i], mu1, sigma1)
                temp2 = (1 - theta) * stats.norm.pdf(self.y[i], mu2, sigma2)
                the[i] = temp1 / (temp1 + temp2)
                print(the[i])
            print('========')
            #M-step:
            print("迭代结果是：")
            ttheta = np.sum(the) / len(self.y)
            mmu1 = np.dot(the, self.y) / np.sum(the)
            mmu2 = np.dot(1 - the, self.y) / np.sum(1 - the)
            ssigma1 = np.dot(the, (self.y - mmu1)**2) / np.sum(the)
            ssigma2 = np.dot(1 - the, (self.y - mmu2)**2) / np.sum(1 - the)
            print(mmu1, mmu2, ssigma1, ssigma2)
            print('========')
            number += 1
            result.append(
                np.array([number, ttheta, mmu1, ssigma1, mmu2, ssigma2]))
            flag = (dist(np.array([theta, mu1, sigma1, mu2, sigma2]),
                         np.array([ttheta, mmu1, ssigma1, mmu2, ssigma2])) >
                    self.epslon)
            theta = ttheta
            mu1 = mmu1
            sigma1 = ssigma1
```

```python
                mu2 = mmu2
                sigma2 = ssigma2
        return result


Y_train = np.array(
    [-67, -48, 6, 8, 14, 16, 23, 24, 28, 29, 41, 49, 56, 60, 75])
"""
假设第一个总体抽了8个(8/15)，第二个总体抽了7个(7/15)
均值1为：-3.000000
方差1为：1047.250000
均值2为：48.285714
方差2为：249.632653
"""

eme = EM_MG(Y_train, 0.533, -3.000000, 1047.250000, 48.285714, 249.632653,
            0.00001)
print(eme.em_estimate())
```

```
0.2093958786536689
0.2093958786536853
0.2093958786536697
0.20939587865375525
========
迭代结果是:
20.933333333315858 20.933333333337966 1329.6622222235571 1329.6622222218687
========
[array([1.00000000e+00, 2.16583788e-01, 2.01553089e+01, 1.37462839e+03,
       2.11484265e+01, 1.31701723e+03]), array([2.00000000e+00, 2.09412683e-01,
2.09318705e+01, 1.32976876e+03,

       2.09337208e+01, 1.32963400e+03]), array([3.00000000e+00, 2.09395918e-01,
2.09333301e+01, 1.32966247e+03,
       2.09333342e+01, 1.32966216e+03]), array([4.00000000e+00, 2.09395879e-01,
2.09333333e+01, 1.32966222e+03,
       2.09333333e+01, 1.32966222e+03]), array([5.00000000e+00, 2.09395879e-01,
2.09333333e+01, 1.32966222e+03,
       2.09333333e+01, 1.32966222e+03])]
```

In [ ]: