深度学习与自然语言处理第二次作业

(EM 算法作业)

姓名：刘千歌　　学号：BY2139121

一、 实验题目

一个袋子中三种硬币的混合比例为：s_1, s_2, 与$1 - s_1 - s_2$ (0<=s_i<=1), 三种硬币掷出正面的概率分别为：p, q, r。（1）自己指定系数 s1, s2, p, q, r，生成 N 个投掷硬币的结果（由 01 构成的序列，其中 1 为正面，0 为反面）;（2）利用 EM 算法来对参数进行估计并与预先假定的参数进行比较。

二、 公式推导

2.1 问题分析与模型建立

为方便表达，设题目中提及的硬币分为 A、B、C 三类，硬币的混合比例（同时代表抽取硬币的概率）表示为π_1、π_2与π_3，正面出现的概率分别是 p, q, r。

进一步，指定 N 为生成样本的数目，生成样本序列表达为$\{x_1, x_2, x_3, \dots, x_N\}$，其中，序列中的数字若为 1，则代表抛硬币结果为正面，数字 0 则代表反面。以第一次投掷为例，若$x_1 = 1$表示可观测到了“硬币出现正面”的结果 ，然而可知道的仅是掷出硬币正面的结果，并不知道是 A、B、C 哪一类硬币的正面。由此，将样本属于哪个类别作为隐变量引入，分别表示为z_1, z_2, z_3（因为我们观测不到结果是从哪个概率分布中得出的，所以将这个叫做隐变量，它一般是离散的）。

2.2 E 步:确定 Q 函数

对于收集到的样本序列，考虑第 i 次观测的x_i，如果投掷的硬币是 A，那么必然伴随着z_1的发生，由此关于x_i的联合概率可以表示为:

$$P(x_i, z_1|\theta) = P(z_1|\theta)P(x_i|z_1, \theta) = \pi_1 p^{x_i}(1-p)^{1-x_i} \quad (1)$$

其他类型同理，再次不赘述。

综上所述，每观测到一枚硬币的投掷结果，观测变量x的概率可以表示为：

$$
\begin{aligned}
P(x|\theta) &= \sum_z P(x,z|\theta) \\
&= \sum_z P(z|\theta)P(x|z,\theta) \\
&= P(z_1)P(y|z_1,\theta) + P(z_2)P(y|z_2,\theta) + P(z_3)P(y|z_3,\theta) \\
&= \pi_1 P(y|z_1,\theta) + \pi_2 P(y|z_2,\theta) + \pi_3 P(y|z_3,\theta) \\
&= p^x(1-p)^{1-x}\pi_1 + q^{x_i}(1-q)^{1-x_i}\pi_2 + r^{x_i}(1-r)^{1-x_i}\pi_3
\end{aligned} \tag{2}
$$

因为 EM 算法是迭代算法，设第 t 次迭代的参数估计值为$\theta^{(t)} = \left(\pi_1^{(t)}, \pi_2^{(t)}, p^{(t)}, q^{(t)}, r^{(t)}\right)$，隐含变量来自于硬币 A、B、C 的后验概率分别表示为 μ_A、μ_B、μ_C。则对于每一个观测变量有以下表示：

$$
\begin{aligned}
\mu_A(x_i) &= \frac{P(z_2, x_i|\theta^{(t)})}{\sum_z P\left(x_i|\theta^{(t)}\right)} \\
&= \frac{p^{x_i}(1-p)^{1-x_i}\pi_1}{p^{x_i}(1-p)^{1-x_i}\pi_1 + q^{x_i}(1-q)^{1-x_i}\pi_2 + r^{x_i}(1-r)^{1-x_i}\pi_3}
\end{aligned} \tag{3}
$$

$$
\begin{aligned}
\mu_B(x_i) &= \frac{P(z_2, x_i|\theta^{(t)})}{\sum_z P\left(x_i|\theta^{(t)}\right)} \\
&= \frac{q^{x_i}(1-q)^{1-x_i}\pi_2}{p^{x_i}(1-p)^{1-x_i}\pi_1 + q^{x_i}(1-q)^{1-x_i}\pi_2 + r^{x_i}(1-r)^{1-x_i}\pi_3}
\end{aligned} \tag{4}
$$

$$
\begin{aligned}
\mu_C(x_i) &= \frac{P(z_3, x_i|\theta^{(t)})}{\sum_z P\left(x_i|\theta^{(t)}\right)} \\
&= \frac{r^{x_i}(1-r)^{1-x_i}\ \pi_3}{p^{x_i}(1-p)^{1-x_i}\pi_1 + q^{x_i}(1-q)^{1-x_i}\pi_2 + r^{x_i}(1-r)^{1-x_i}\pi_3} \\
&= 1 - \mu_A(x_i) - \mu_B(x_i)
\end{aligned} \tag{5}
$$

上述后验概率是一个常量，因为它是基于过去的估计量以及真实观测数据计算得到。进一步，写出 Q 函数公式表达式，并把上文得出的公式(1)、(3)、(4)、(5)结论带入 Q 函数的表达式。

$$
\begin{aligned}
Q(\theta,\theta^{(t)}) &= \sum_{Z} P(\boldsymbol{Z}|X,\theta^{(t)}) \log P(X,\boldsymbol{Z}|\theta) \\
&= \sum_{Z} \log P(X,\boldsymbol{Z}|\theta)^{P(\boldsymbol{Z}|X,\theta^{(t)})} \\
&= \sum_{Z}\sum_{i=1}^{N} \log P(x_i,\boldsymbol{Z}|\theta)^{P(\boldsymbol{Z}|x_i,\theta^{(t)})} \\
&= \sum_{i=1}^{N} \{P(z_1|x_i,\theta^{(t)}) \log P(x_i,z_1|\theta) + P(z_2|x_i,\theta^{(t)}) \log P(x_i,z_2|\theta) + P(z_3|x_i,\theta^{(t)}) \log P(x_i,z_3|\theta) \\
&= \sum_{i=1}^{N} \{\mu_A(x_i) \log P(x_i,z_1|\theta) + \mu_B(x_i) \log P(x_i,z_2|\theta) + \mu_C(x_i) \log P(x_i,z_3|\theta) \\
&= \sum_{i=1}^{N} \{\mu_A(x_i) \log \pi_1 p^{x_i}(1-p)^{1-x_i} + \mu_B(x_i) \log \pi_2 q^{x_i}(1-q)^{1-x_i} + \mu_C(x_i) \log \pi_3 r^{x_i}(1-r)^{1-x_i}
\end{aligned} \tag{6}
$$

2.3 M 步:更新参数

根据以上内容得出求取$\theta^{(t+1)}$的目标函数:

$$
\theta^{(t+1)} = argmax_{\theta} Q(\theta,\theta^{(t)}) \tag{7}
$$

2.3.1 求取(π_1,π_2,π_3)

然后求得$Q(\theta,\theta^{(t)})$偏导可得参数的极大似然估计，推导过程如下所示：

$$
\begin{aligned}
\frac{\partial Q(\theta,\theta^{(t)})}{\partial \pi_1^{(t+1)}} &= \sum_{i=1}^{N} \{\mu_A^{(t)}(x_i) \frac{p^{x_i}(1-p)^{1-x_i}}{\pi_1^{(t+1)} p^{x_i}(1-p)^{1-x_i}} - \mu_C^{(t)}(x_i) \frac{r^{x_i}(1-r)^{1-x_i}}{(1-\pi_1^{(t+1)}-\pi_2^{(t+1)}) r^{x_i}(1-r)^{1-x_i}} \} \\
&= \sum_{i=1}^{N} \left\{ \frac{\mu_A^{(t)}(x_i)}{\pi_1^{(t+1)}} - \frac{\mu_C^{(t)}(x_i)}{1-\pi_1^{(t+1)}-\pi_2^{(t+1)}} \right\} \\
&= 0
\end{aligned} \tag{8}
$$

$$
\begin{aligned}
\frac{\partial Q(\theta,\theta^{(t)})}{\partial \pi_2^{(t+1)}} &= \sum_{i=1}^{N} \{\mu_B^{(t)}(x_i) \frac{p^{x_i}(1-p)^{1-x_i}}{\pi_1^{(t+1)} p^{x_i}(1-p)^{1-x_i}} - \mu_C^{(t)}(x_i) \frac{r^{x_i}(1-r)^{1-x_i}}{(1-\pi_2^{(t+1)}-\pi_3^{(t+1)}) r^{x_i}(1-r)^{1-x_i}} \} \\
&= \sum_{i=1}^{N} \left\{ \frac{\mu_B^{(t)}(x_i)}{\pi_2^{(t+1)}} - \frac{\mu_C^{(t)}(x_i)}{1-\pi_2^{(t+1)}-\pi_3^{(t+1)}} \right\} \\
&= 0
\end{aligned} \tag{9}
$$

根据公式(8)、（9）进行变换，可得公式(10)、(11)形式，

$$
\frac{\pi_1^{(t+1)}}{\left(1-\pi_1^{(t+1)}-\pi_2^{(t+1)}\right)} = \frac{\sum_{i=1}^{N} \mu_A^{(t)}(x_i)}{\sum_{i=1}^{N} \mu_C^{(t)}(x_i)} \tag{10}
$$

$$
\frac{\pi_2^{(t+1)}}{(1-\pi_1^{(t+1)}-\pi_2^{(t+1)})} = \frac{\sum_{i=1}^{N} \mu_B^{(t)}(x_i)}{\sum_{i=1}^{N} \mu_C^{(t)}(x_i)} \tag{11}
$$

将公式(10)、(11)相比，最终得到由π_1表示的π_2表达式(13)。

$$\frac{\pi_2^{(t+1)}}{\pi_1^{(t+1)}} = \frac{\sum_{i=1}^{N}\mu_B^{(t)}(x_i)}{\sum_{i=1}^{N}\mu_A^{(t)}(x_i)} \tag{12}$$

$$\pi_2^{(t+1)} = \frac{\sum_{i=1}^{N}\mu_B^{(t)}(x_i)\,\pi_1^{(t+1)}}{\sum_{i=1}^{N}\mu_A^{(t)}(x_i)} \tag{13}$$

将(13)带入公式(8)，得到公式如下：

$$\sum_{i=1}^{N}\mu_A^{(t)}(x_i)\left(1-\pi_1^{(t+1)}-\pi_2^{(t+1)}\right) = \pi_1^{(t+1)}\sum_{i=1}^{N}\left\{1-\mu_A^{(t)}(x_i)-\mu_B^{(t)}(x_i)\right\} \tag{14}$$

对上式进行消元，得到公式(15)：

$$\sum_{i=1}^{N}\mu_A^{(t)}(x_i) = N\pi_1^{(t+1)} \tag{15}$$

进一步得到π_1的估计值，并带入公式(11)获取π_2的估计值：

$$\pi_1^{(t+1)} = \frac{\sum_{i=1}^{N}\mu_A^{(t)}(x_i)}{N} \tag{16}$$

$$\pi_2^{(t+1)} = \frac{\sum_{i=1}^{N}\mu_B^{(t)}(x_i)}{\sum_{i=1}^{N}\mu_A^{(t)}(x_i)}\cdot\frac{\sum_{i=1}^{N}\mu_A^{(t)}(x_i)}{N} = \frac{\sum_{i=1}^{N}\mu_B^{(t)}(x_i)}{N} \tag{17}$$

π_3的估计值如下:

$$\pi_3^{(t+1)} = 1-\pi_2^{(t+1)}-\pi_1^{(t+1)} \tag{18}$$

2.3.2 求取(p,q,r)

进一步，求取p,q,r的偏导，则有公式如下：

$$\begin{aligned}\frac{\partial Q(\theta,\theta^{(t)})}{\partial \mathrm{p}^{(t+1)}} &= \sum_{i=1}^{N}\mu_A(x_i)\frac{x_i p^{x_i-1}(1-p)^{1-x_i}+(x_i-1)p^{x_i}(1-p)^{-x_i}}{p^{x_i}(1-p)^{1-x_i}} \\ &= \sum_{i=1}^{N}\mu_A(x_i)\left(\frac{x_i}{p}+\frac{(x_i-1)}{1-p}\right) \\ &= \sum_{i=1}^{N}\mu_A(x_i)\left(\frac{x_i(1-p)+p(x_i-1)}{p(1-p)}\right) \\ &= \sum_{i=1}^{N}\mu_A(x_i)\left(\frac{x_i-p}{p(1-p)}\right) = 0\end{aligned} \tag{19}$$

公式(19)可简化如下：

$$\sum_{i=1}^{N}\mu_A(x_i)x_i-\sum_{i=1}^{N}\mu_A(x_i)p=0 \tag{20}$$

对(20)进行变换，可得$p^{(t+1)}$的估计量。

$$p^{(t+1)}=\frac{\sum_{i=1}^{N}\mu_A(x_i)x_i}{\sum_{i=1}^{N}\mu_A(x_i)} \tag{21}$$

同理可得$q^{(t+1)}$与$r^{(t+1)}$的估计量，如公式(22)与(23)所示。

$$q^{(t+1)}=\frac{\sum_{i=1}^{N}\mu_B(x_i)x_i}{\sum_{i=1}^{N}\mu_B(x_i)} \tag{22}$$

$$r^{(t+1)}=\frac{\sum_{i=1}^{N}\mu_C(x_i)x_i}{\sum_{i=1}^{N}\mu_C(x_i)} \tag{23}$$

本轮迭代更新参数$\theta^{(t+1)}=\left(\pi_1^{(t+1)},\pi_2^{(t+1)},p^{(t+1)},q^{(t+1)},r^{(t+1)}\right)$。若未达成收敛，则进入新一轮迭代，将其再次带入 Q 函数。

三、 代码说明

首先，依据下方入口函数的代码片段说明代码的整体逻辑。样本生成长度 N 以及算法最大迭代次数 iter 可根据需求进行更改。进一步，设定对照参数 `paras` 并传入 `generate_dataset` 函数生成样本，初始化用于传入 EM 算法的初始参数$\theta^{(0)}=\left(\pi_1^{(0)},\pi_2^{(0)},p^{(0)},q^{(0)},r^{(0)}\right)$（这里以$(0.4,0.5,0.6,0.7,0.3)$为例）。最终 EM 计算得出参数结果会与 `paras` 进行对比，送入 `gen_picture` 函数生成可视化对比图。

```
if __name__ =="__main__":
    #生成样本序列长度
    N = 1000
    #允许的最大迭代次数
    iter = 100
    #初始化用于生成样本集的对照参数
    paras = [_pi1, _pi2, _p, _q, _r]=[0.3, 0.4, 0.6, 0.4,
0.7]
```

```
# 初始化 pi1，pi2,p,q,r 参数
[pi1,pi2,p,q,r] = [0.4,0.5,0.6,0.7,0.3]
#以此为依据，生成样本集
x = generate_dataset(0.4,0.5,0.6,0.7,0.3,N)
# 将参数与样本数据集送入 em 算法
em(x,pi1,pi2,p,q,r,100)
gen_picture(paras)
```

代码段 1 代码总逻辑

依据 paras 参数生成样本的过程如代码段 2 所示。具体逻辑已标注。最终返回 0，1 组成的数组序列。1 代表正面，0 代表反面。

```
def generate_dataset(pi1,pi2,p,q,r,N):
    #初始化样本序列
    sequences = []
    for i in range(N):
        #选定(0,1)范围内的一个随机数
        choice_seed = random.random()
        sequence = []

        #若随机数小于落在 0 与 pi1 之间，则选择硬币 A
        if choice_seed >=0 and choice_seed <pi1:
          #选定硬币 A 后投掷 M 次
          for i in range(M):
              #再次选择 0 到 1 间的随机数 status
              status = random.random()
               #若 status 小于概率 p 则生成反面样本，反之生成正面样本
              if status >=0 and status <= p:
                  sequence.append(1)
              else:
                 sequence.append(0)
            sequences .append(sequence)
       #若随机数小于落在 pi1 与 pi1+pi2 之间，则选择硬币 B
        if choice_seed >=pi1 and choice_seed < pi1+pi2:
              for i in range(M):
```

```
            status = random.random()
            if status >=0 and status <= q:
                sequence.append(1)
            else:
               sequence.append(0)
        sequences.append(sequence)

    #若随机数小于落在 pi1+pi2 与 1 之间，则选择硬币 C
        if choice_seed >= pi1+pi2 and choice_seed < 1:
           for i in range(M):
               status = random.random()
               if status >= 0 and status <= r:
                sequence.append(1)
               else:
                 sequence.append(0)
          sequencecs.append(sequence)
    #返回生成的样本序列
    return sequences
```

代码段 2 生成样本数据集

EM 算法以初始对照参数作为输入，在经过 e_step 获取后验概率后，输入 m_step 进行参数的更新，若参数无更新则认定收敛。

```
def em (observed_x, start_pi1,start_pi2,start_p,
start_q, start_r, iter_num):
    """
    :param observed_x:   观测数据
    :param start_pi1:   下一次迭代开始的 pi1
    :param start_pi2:   下一次迭代开始的 pi2
    :param start_p:   下一次迭代开始的 p
    :param start_q:   下一次迭代开始的 q
    :param start_q:   下一次迭代开始的 r
    :param iter_num:   迭代次数
    :return:
    """
    #开启迭代
    for i in range(iter_num):
       #传入本轮初始参数到 e_step，分别计算三类硬币的后验概率μ_A,μ_B,μ_C
        mu_a,mu_b,mu_c =e_step(start_pi1, start_pi2,
start_p, start_q,start_r,observed_x)
```

```
        #打印本轮初始参数
        print ("第"+str(i)+"轮: ","参数: ",
        [start_pi1,start_pi2,start_p,start_q,start_r])

        #送入 m_step 更新参数，若两轮参数一致则认定收敛停止迭代
        if [start_pi1, start_pi2, start_p, start_q,
    start_r] ==m_step(mu_a,mu_b,mu_c, observed_x):
            break
        else:
         [start_pi1,start_pi2,start_p,start_q,start_r]=
         m_step(mu_a, mu_b, mu_c, observed_x)
```

代码段 3 EM **算法的整体流程**

接下来是 e_step 的代码，其后验概率计算步骤与前文的公式推导保持一致，唯一不同的是每一轮迭代投掷次数 M 的加入。

```
def e_step(pi1,pi2,p,q,r,x):
    """
    e 步计算的单次迭代
    :param pi1: 下一次迭代开始的 pi1
    :param pi2: 下一次迭代开始的 pi2
    :param p:  下一次迭代开始的 p
    :param q:  下一次迭代开始的 q
    :param x: 观察数据
    :return:
    """
    pi3 = 1- pi1-pi2
    mu_a = []
    mu_b = []
    mu_c = []
    for xi in x:
        # 求 mu_a
        a_item = pi1 * math.pow(p, sum(xi)) * math.pow(1 -
p, M - sum(xi)) / \
        float(pi1 * math.pow(p, sum(xi)) * math.pow(1 - p,
M - sum(xi)) + pi2 * math.pow(q, sum(xi)) * math.pow(1 -
q, M- sum(xi))+ pi3 * math.pow(r, sum(xi)) * math.pow(1 -
r, M - sum(xi)))
        mu_a.append(a_item)
        #求 mu_b
        b_item = pi2 * math.pow(q, sum(xi)) * math.pow(1 -
```

```
q, M - sum(xi)) / \
        float(pi1 * math.pow(p, sum(xi)) * math.pow(1 - p,
M - sum(xi)) +
            pi2 * math.pow(q, sum(xi)) * math.pow(1 - q, M -
sum(xi)) + pi3 * math.pow(r, sum(xi)) * math.pow(1 - r, M
- sum(xi)))
        mu_b.append(b_item)

        #求 mu_c
        c_item = pi3 * math.pow(r, sum(xi)) * math.pow(1 -
r, M - sum(xi)) / \
        float(pi1 * math.pow(p, sum(xi)) * math.pow(1 - p,
M - sum(xi)) +
            pi2 * math.pow(q, sum(xi)) * math.pow(1 - q, M -
sum(xi)) + pi3 * math.pow(r, sum(xi)) * math.pow(1 - r, M
- sum(xi)))
        mu_c.append(c_item)
    return mu_a,mu_b,mu_c
```

代码段 3 EM **算法的** e-step

接下来是 q_step 的代码，用于参数的更新。计算步骤与前文的公式推导保持一致，唯一不同的是每一轮迭代投掷次数 M 的加入。

```
def m_step(mu_a,mu_b,mu_c,x):
    """
     m 步计算
    :param mu: e-step 获取的后验概率
    :param x:  观察数据
    :return:
    """
    #更新参数π1、π2
    new_pi1 = sum(mu_a)/len(mu_a)
    new_pi2 = sum(mu_b)/len(mu_b)
    #更新参数p,q,r
    new_p = sum([mu_a[i] * sum(x[i]) for i in
range(len(mu_a))]) / sum(mu_a[i] * M for i in
range(len(mu_a)))
    new_q = sum([mu_b[j]* sum(x[j]) for j in
range(len(mu_b))]) / sum(mu_b[j] * M for j in
range(len(mu_b)))
    new_r = sum([mu_c[h] * sum(x[h]) for h in
range(len(mu_c))]) / sum(mu_c[h] * M for h in
range(len(mu_c)))
    return [new_pi1, new_pi2, new_p,new_q,new_r]
```

代码段 4 EM **算法的** m-step

四、 结果分析

本章节将分别针对三个问题设计实验并对其结果进行深入分析：

- 收敛效果与样本序列长度 N 的关系？
- 收敛效果与单轮抽取硬币投掷次数 M 的关系？
- 收敛效果与初始化参数的关系？

4.1 收敛效果与样本序列长度 N 的关系

实验背景：指定对照参数为（0.3，0.4，0.6，0.4，0.7），单轮投掷次数 M 为 100 生成训练样本；初始化 $\theta^{(0)} = \left(\pi_1^{(0)}, \pi_2^{(0)}, p^{(0)}, q^{(0)}, r^{(0)}\right) = (0.2,0.4,0.5,0.3,0.6)$作为 EM 算法首轮迭代参数，指定最大迭代次数为 50 次。

实验设计：分别选定 10、100、1000、10000 作为样本长度 N，通过图例表示参数 EM 算法的收敛效果。

实验结果：

N=10 的情况如下图所示。左边的图表示π_1、π_2与π_3三个参数的收敛情况，横轴为迭代次数，纵轴为参数的值，不同颜色所代表的参数已在图上标出；右侧的图为p、q与r的收敛情况，两张图中的虚线标注了对照参数的大小。后文中的图片与以上陈述含义一致，之后不再赘述。

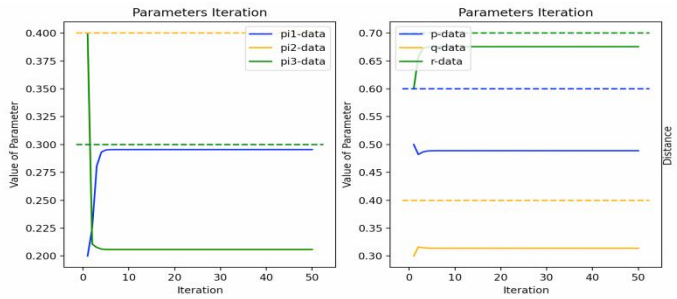

图 1 N= 10

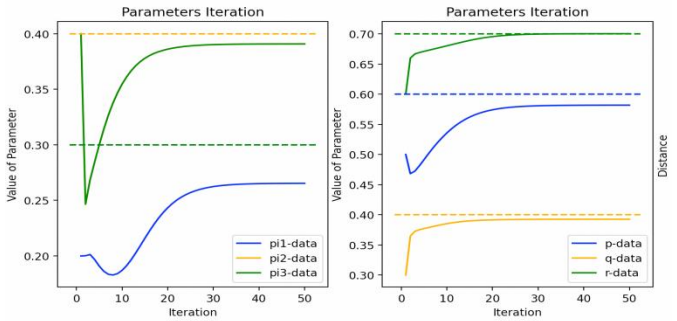

图 2 N = 100

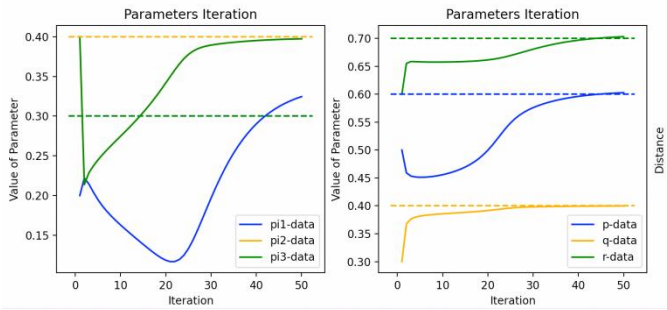

图 3 N =1000

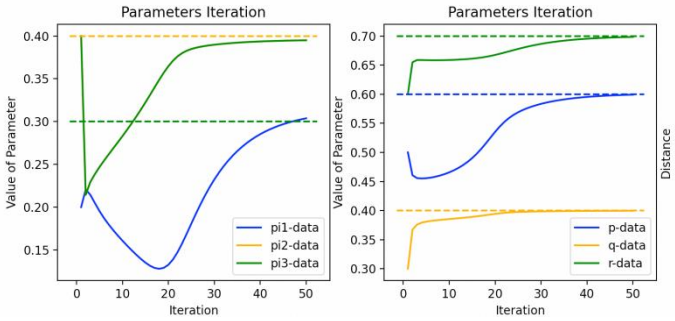

图 4 N=10000

实验分析：从图中可以看出，在 N=10 时，参数快速收敛至极值，单距离对照值距离较大。随着样本长度的增加，算法的收敛效果增强，其原因可能是大基数样本使生成样本的对照参数有更大的体现面。在 N=1000 以及 N=10000 时 p、q、r 最终已与对照参数一致。

4.2 收敛效果与单轮抽取硬币投掷次数 M 的关系

实验背景：指定对照参数为（0.3，0.4，0.6，0.4，0.7），规定样本投掷次数 N 为 1000；初始化$\theta^{(0)} = \left(\pi_1^{(0)}, \pi_2^{(0)}, p^{(0)}, q^{(0)}, r^{(0)}\right) = (0.2,0.4,0.5,0.3,0.6)$作为 EM 算法首轮迭代参数，指定最大迭代次数为 50 次。

实验设计：分别选定 1、10、100、200 作为单轮投掷次数 M，通过图例表示参数 EM 算法的收敛效果。

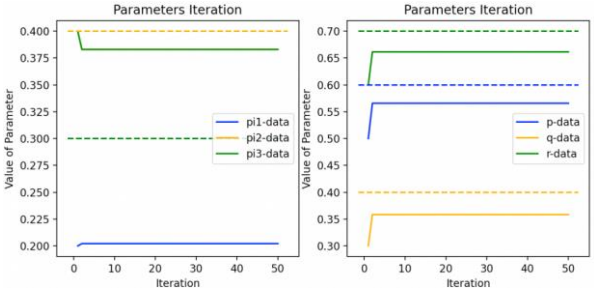

图 5 M = 1

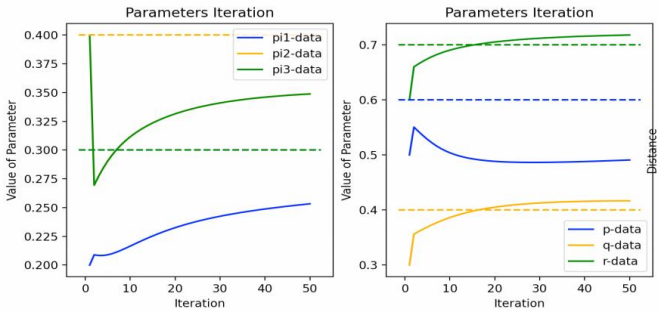

图 6 M = 10

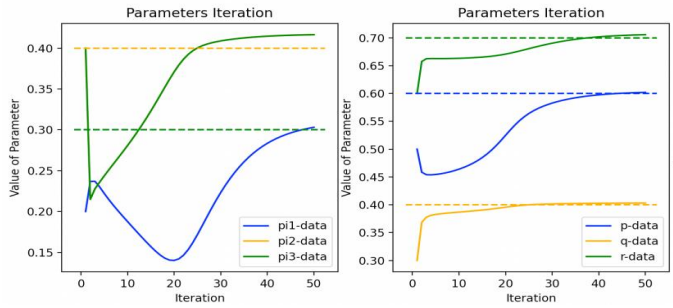

图 7 M = 100

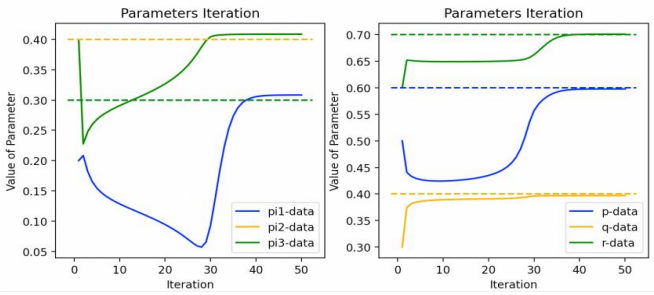

图 8 M = 200

实验分析：相同条件下，M 越大收敛的效果越好，p、q、r 在 M=100 和 M=200 时最终能收敛至初始设定对照值。其原因可能是，M 越大，最终生成样本就有越大几率体现初始用于生成样本数据的对照参数的特征。

4.3 收敛效果与初始化参数的关系

实验背景：依然指定对照参数为（0.3，0.4，0.6，0.4，0.7），规定样本投掷次数 N 为 1000，单轮投掷次数 M 为 100，指定最大迭代次数为 50 次。

实验设计：分别选用与初始参数偏移量依次递减的初始化参数进行 EM 计算，并用图例展示收敛效果。

1)初始化$\theta^{(0)} = \left(\pi_1^{(0)}, \pi_2^{(0)}, p^{(0)}, q^{(0)}, r^{(0)}\right) = (0.2,0.4,0.3,0.2,0.5)$作为 EM 算法首轮迭代参数，

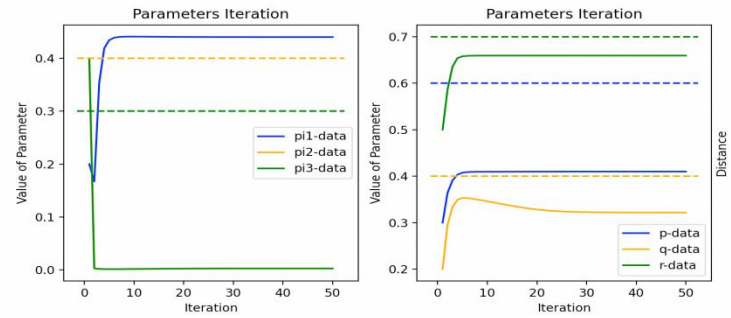

图 9 大偏移量初始向量收敛效果

2)初始化$\theta^{(0)}=\left(\pi_1^{(0)},\pi_2^{(0)},p^{(0)},q^{(0)},r^{(0)}\right)=(0.2,0.4,0.4,0.3,0.6)$作为 EM 算法首轮迭代参数。

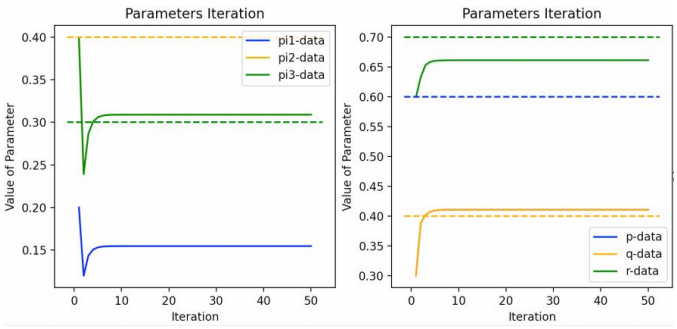

图 10 较小偏移量初始向量收敛效果

3)初始化$\theta^{(0)}=\left(\pi_1^{(0)},\pi_2^{(0)},p^{(0)},q^{(0)},r^{(0)}\right)=(0.2,0.4,0.5,0.4,0.6)$作为 EM 算法首轮迭代参数。

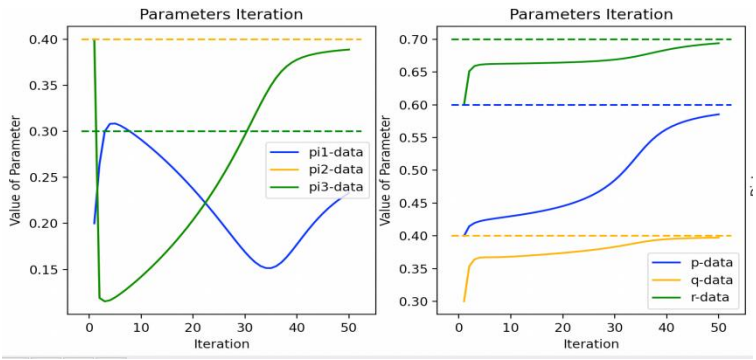

图 11 小偏移量初始向量收敛效果

实验分析：EM 算法又称为期望最大化算法，其本质上是非凸的。十分容易陷入局部最优的局面。而参数 θ 的选择直接影响收敛效率以及能否得到全局最优解，一如上图所示，收敛结果对初始值的选择十分敏感。

总的来说，EM 算法收敛的优劣很大程度上取决于其初始参数。