姓名：陈真　　学号：SY2103801

1.背景知识

EM 算法即期望最大化算法(expection maximization algorithm)是一种迭代算法，作为一种数据添加算法，在目前的 DL 算法中被广泛运用。EM 算法推导如下。

a.数据集

观测数据：观测到的随机变量 X 的样本：X = (x1,..., xn)

隐含变量：未观测到的随机变量 Z 的值：Z = (z1,..., zn)

完整数据：包含观测到的随机变量 X 和隐含变量 Z 的数据：Y = (X, Z)

b.EM 算法的推导

EM 算法是从含有隐含变量的数据(完整数据)中计算极大似然估计。Z 为隐含变量，则从可观测数据入手，对参数进行极大似然估计。

根据边缘分布列的定义：

$$\sum_{i=1}^{+\infty} P\left(X = x_i, Y = y_j\right) = P\left(X = x_i\right)$$

首先改写 $L(\theta)$：

$$L(\theta) = \sum_i \ln p\left(x^{(i)};\theta\right) = \sum_i \ln \sum_{z^{(i)}} p\left(x^{(i)}, z^{(i)};\theta\right)$$

上式中将 $x^{(i)}$ 用边缘分布列反向拆解为联合分布。

接着，定义隐含变量 Z 的分布的分布 Q_i。Q_i 表示隐含变量 Z 的某种分布，且：

$$\sum_z Q_i\left(z^{(i)}\right) = 1 \quad Q_i(z) \geq 0$$

于是 $L(\theta)$ 可以改写成：

$$L(\theta) = \sum_i \ln \sum_{z^{(i)}} Q_i\left(z^{(i)}\right) \frac{p\left(x^{(i)}, z^{(i)};\theta\right)}{Q_i\left(z^{(i)}\right)}$$

利用 jensen 不等式，对于凹函数 $f(x) = \ln x$，有 $\ln(E[X]) \geq E[\ln X]$。因此：

$$L(\theta) = \sum_i \ln \sum_{z^{(i)}} Q_i\left(z^{(i)}\right) \frac{p\left(x^{(i)}, z^{(i)};\theta\right)}{Q_i\left(z^{(i)}\right)} = \sum_i \ln\left(E\left[\frac{p\left(x^{(i)}, z^{(i)};\theta\right)}{Q_i\left(z^{(i)}\right)}\right]\right) \geq \sum_i E\left[\ln \frac{p\left(x^{(i)}, z^{(i)};\theta\right)}{Q_i\left(z^{(i)}\right)}\right]$$

$$\sum_i E\left[\ln \frac{p\left(x^{(i)}, z^{(i)};\theta\right)}{Q_i\left(z^{(i)}\right)}\right] = \sum_i \sum_{z^{(i)}} Q_i\left(z^{(i)}\right) \ln \frac{p\left(x^{(i)}, z^{(i)};\theta\right)}{Q_i\left(z^{(i)}\right)}$$

所以：

$$L(\theta) \geq \sum_i \sum_{z^{(i)}} Q_i\left(z^{(i)}\right) \ln \frac{p\left(x^{(i)}, z^{(i)};\theta\right)}{Q_i\left(z^{(i)}\right)}$$

固定$\theta^{(t)}$，调整$Q(z)$使下界上升，至与此$\theta^{(t)}$对应的$L\left(\theta^{(t)}\right)$相等。然后固定$Q(z)$，利用极大似然估计调整$\theta$使得下届达到最大值，得到的新的$\theta$记为$\theta^{(t+1)}$。依次重复上述步骤。

c.等号成立条件

现在考察等式成立的条件。等号成立时知:

$$\frac{p\left(x^{(i)}, z^{(i)} ; \theta\right)}{Q_{i}\left(z^{(i)}\right)}=C$$

经过变换上式得到:

$$\begin{aligned}
& \frac{p\left(x^{(i)}, z^{(i)} ; \theta\right)}{Q_{i}\left(z^{(i)}\right)}=C \\
& \Rightarrow P\left(x^{(i)}, z^{(i)} ; \theta\right)=C\left(Q_{i}\left(z^{(i)}\right)\right) \\
& \Rightarrow \sum_{z} P\left(x^{(i)}, z^{(i)} ; \theta\right)=C\left(\sum_{z} Q_{i}\left(z^{(i)}\right)\right) \\
& \Rightarrow \sum_{z} P\left(x^{(i)}, z^{(i)} ; \theta\right)=C
\end{aligned}$$

那么:

$$\begin{aligned}
& Q_{i}\left(z^{(i)}\right) \\
& =\frac{p\left(x^{(i)}, z^{(i)} ; \theta\right)}{\sum_{z} p\left(x^{(i)}, z^{(i)} ; \theta\right)} \\
& =\frac{p\left(x^{(i)}, z^{(i)} ; \theta\right)}{p\left(x^{(i)} ; \theta\right)} \\
& =p\left(z^{(i)} \mid x^{(i)} ; \theta\right)
\end{aligned}$$

从概率的角度而言，$Q_{i}\left(z^{(i)}\right)$表示为在$\theta$参数的模型中，在$x^{(i)}$的条件下，取到$z^{(i)}$的概率。

d.迭代步骤:

E-step:固定θ，得到$Q_{i}\left(z^{(i)}\right)$的计算公式$Q_{i}\left(z^{(i)}\right)=p\left(z^{(i)} \mid x^{(i)} ; \theta\right)$，同时建立$L(\theta)$的下届表达式:

$$L(\theta)=\sum_{i} \sum_{z^{(i)}} Q_{i}\left(z^{(i)}\right) \ln \frac{p\left(x^{(i)}, z^{(i)} ; \theta\right)}{Q_{i}\left(z^{(i)}\right)}$$

M-step：用极大似然计算θ，

$$\theta:=\arg \max _{\theta} \sum_{i} \sum_{z^{(i)}} Q_{i}\left(z^{(i)}\right) \ln \frac{p\left(x^{(i)}, z^{(i)} ; \theta\right)}{Q_{i}\left(z^{(i)}\right)}$$

2.题目及实现过程

a.题目:

一个袋子中三种硬币的混合比例为：s1，s2 与 1-s1-s2 (0<=si<=1)，三种硬币掷出正面的概率分别为：p, q, r。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），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。

b.数据生成:

假设 $s1 = 0.3, s2 = 0.2, p = 0.3, q = 0.4, r = 0.5$,按照此概率分布去采样生成 N 个投掷结果。定义 1 为正面，0 为反面，得到结果序列。

c.变量定义:

根据上述的分析，知 $\theta =< s1, s2, p, q, r >$，其中通过实验已知变量 X 为硬币的面的正反，隐藏变量 Z 为硬币的种类。

d.初始化:

初始假设 $\hat{\theta}_{(0)} =< 0.5, 0.1, 0.4, 0.5, 0.6 >$

e.E-step:

对于第 e 次迭代的第 i 个 x，算出来的隐含变量 z1，z2 表达式如下:

$$z1_{(e)}(x^{(i)}) = \frac{p_{(e-1)}^{x^{(i)}}(1-p_{(e-1)})^{1-x^{(i)}}\, s1_{(e-1)}}{p_{(e-1)}^{x^{(i)}}(1-p_{(e-1)})^{1-x^{(i)}}\, s1_{(e-1)} + q_{(e-1)}^{x^{(i)}}(1-q_{(e-1)})^{1-x^{(i)}}\, s2_{(e-1)} + r_{(e-1)}^{x^{(i)}}(1-r_{(e-1)})^{1-x^{(i)}}\,(1-s1_{(e-1)}-s2_{(e-1)})}$$

$$z2_{(e)}(x^{(i)}) = \frac{q_{(e-1)}^{x^{(i)}}(1-q_{(e-1)}^{x^{(i)}})^{1-x^{(i)}}\, s2_{(e-1)}}{p_{(e-1)}^{x^{(i)}}(1-p_{(e-1)})^{1-x^{(i)}}\, s1_{(e-1)} + q_{(e-1)}^{x^{(i)}}(1-q_{(e-1)})^{1-x^{(i)}}\, s2_{(e-1)} + r_{(e-1)}^{x^{(i)}}(1-r_{(e-1)})^{1-x^{(i)}}\,(1-s1_{(e-1)}-s2_{(e-1)})}$$

f.M-step:

对于第 e 次迭代，根据极大似然估计的 $\hat{\theta}_{(e)}$ 为

$$\hat{\theta}_{(e)}[0] = s1_{(e)} = \left(\sum_{i=1}^{i=N} z1_{(e)}(x^{(i)})\right) / N$$

$$\hat{\theta}_{(e)}[1] = s2_{(e)} = \left(\sum_{i=1}^{i=N} z2_{(e)}(x^{(i)})\right) / N$$

$$\hat{\theta}_{(e)}[2] = p_{(e)} = \left(\sum_{i=1}^{i=N}\left(x^{(i)} \bullet z1_{(e)}(x^{(i)})\right)\right) / \left(\sum_{i=1}^{i=N} z1_{(e)}(x^{(i)})\right)$$

$$\hat{\theta}_{(e)}[3] = q_{(e)} = \left(\sum_{i=1}^{i=N}\left(x^{(i)} \bullet z2_{(e)}(x^{(i)})\right)\right) / \left(\sum_{i=1}^{i=N} z2_{(e)}(x^{(i)})\right)$$

$$\hat{\theta}_{(e)}[4] = r_{(e)} = \left(\sum_{i=1}^{i=N}\left(x^{(i)} \bullet \left(1 - z1_{(e)}(x^{(i)}) - z2_{(e)}(x^{(i)})\right)\right)\right) / \left(N - \sum_{i=1}^{i=N} z1_{(e)}(x^{(i)}) - \sum_{i=1}^{i=N} z2_{(e)}(x^{(i)})\right)$$

g.迭代顺序:

迭代顺序如图 1 所示:

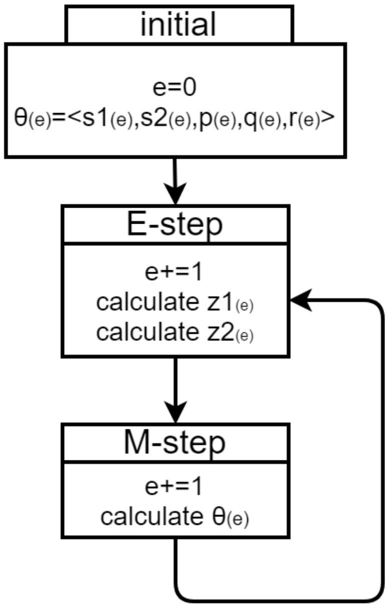

图 1 迭代顺序

3.实验分析

a.实验计划：

Exp1：按照 b 进行数据生成，取 N=50，100，200，250. 按照 d 中进行 $\hat{\theta}$ 的初始化。

$\hat{\theta}_{(e)}$ 随迭代次数 e 的曲线如下.

Exp2：改变 d 中进行 $\hat{\theta}$ 的初始化的值，对比初值敏感度。

b.实际实验：

实验代码见附页。选取 10 次初值，N 长度选取最大至 10000，迭代次数选取最大至 5000. 都无法收敛到给定初值。基本收敛到假定的初值附近。分析：收敛到局部解。给定条件较少。可以采取的做法同时抛一枚硬币数次，但是不知道硬币是哪种。

4.实验代码

```
#nlp-2,2022-4-19,by 陈真 SY2103801
import numpy
import xlwt
#数据生成
def generate(N,s1s2pqr,seed):
    numpy.random.seed(seed)
    X=[[] for i in range(len(N))]
    for g,en in enumerate(N):
        for i in range(en):#50
            #抛种类
            test = numpy.random.choice(numpy.arange(0, 3),
p=[s1s2pqr[0], s1s2pqr[1],1-s1s2pqr[0]-s1s2pqr[1]])
            #抛正反
            if test==0:
```

```
                X[g].append(int(numpy.random.choice(numpy.arange(0, 2),
p=[1-s1s2pqr[2], s1s2pqr[2]])))
            if test==1:
                X[g].append(int(numpy.random.choice(numpy.arange(0, 2),
p=[1-s1s2pqr[3], s1s2pqr[3]])))
            if test==2:
                X[g].append(int(numpy.random.choice(numpy.arange(0, 2),
p=[1-s1s2pqr[4], s1s2pqr[4]])))
    return X

s1s2pqr=[0.3,0.2,0.3,0.4,0.5]#给定的比例
N=[50,100,200,250]#不同的序列长度
seed=5#
X=generate(N,s1s2pqr,seed)

#初始化
theta0=[0.1,0.2,0.4,0.5,0.6]
book = xlwt.Workbook(encoding='utf-8',style compression=0)
sheet = book.add sheet('exp2',cell overwrite ok=True)
col =
('s1 N0','s2 N0','p N0','q N0','r N0','s1 N1','s2 N1','p N1','q N1','
r N1','s1 N2','s2 N2','p N2','q N2','r N2','s1 N3','s2 N3','p N3','q
N3','r N3')
for i in range(0,len(col)):
      sheet.write(0,i,col[i])

def zfunction(theta,X,st):
    zz=[]
    if st=='z1':
        for x in X:
            zmeta=(theta[0]*(theta[2]**x)*(1-theta[2])**(1-x))/\
                  (theta[0]*theta[2]**x*(1-theta[2])**(1-x)+
                   theta[1]*theta[3]**x*(1-theta[3])**(1-x)+
                   (1-theta[1]-theta[0])*theta[4]**x*(1-theta[4])**(1-
x))
            zz.append(zmeta)
    else:
        for x in X:
            zmeta = (theta[1] * theta[3] ** x * (1 - theta[3]) ** (1 -
x)) / (
                        theta[0] * theta[2] ** x * (1 - theta[2]) ** (1 -
x) + \
                        theta[1] * theta[3] ** x * (1 - theta[3]) ** (1 -
x) + \
                        (1 - theta[1] - theta[0]) * theta[4] ** x * (1 -
theta[4]) ** (1 - x))
            zz.append(zmeta)

    return zz

Th=[theta0 for i in range(len(N))]
for epoch in range(10):
    print(Th)
    # E-step
    Z1=[[] for i in range(len(N))]
    Z2=[[] for i in range(len(N))]
    for j,z1 in enumerate(Z1):
        z1+=zfunction(Th[j],X[j],'z1')
    for j,z2 in enumerate(Z2):
        z2+=zfunction(Th[j],X[j],'z2')
    print('Z1',Z1)
    print('Z2',Z2)
    # M-step
```

```
    for nu,the in enumerate(Th):
        the[0]=sum(Z1[nu])/N[nu]
        the[1]=sum(Z2[nu])/N[nu]
        the[2]=sum([Z1[nu][i]*X[nu][i] for i in
range(len(Z1[nu]))])/sum(Z1[nu])
        the[3]=sum([Z2[nu][i]*X[nu][i] for i in
range(len(Z2[nu]))])/sum(Z2[nu])
        the[4]=sum([(1-Z1[nu][i]-Z2[nu][i])*X[nu][i] for i in
range(len(Z2[nu]))])/(N[nu]-sum(Z1[nu])-sum(Z2[nu]))
    print(Th)
    print('---------------------')
    for k,eve in enumerate(Th):
        for j in range(len(eve)):
            sheet.write(epoch + 1, 5*k+j, eve[j])

savepath = './excel表格.xls'
book.save(savepath)
```