MLE,EM,EM acceleration

Guanting Wei

Nov.3.2018

1 Abstract

In statistics, maximum likelihood estimation (MLE) is a method of estimating the parameters of a statistical model, given observations. However, sometimes we can not use MLE due to the missing data or latent variable. In this situation, EM algorithm works. Further more, in order to improve EM algorithm, we can use EM acceleration. In other words, Em algorithm can be recognized a special situation of MLE.

2 Sample

Here we will use a sample to learn EM and use some methods to improve it. Suppose that there are 4 kinds of animals(A,B,C,D) in a farm. And their probability are

$$P(A) = \frac{1}{2} + \frac{\theta}{4}$$

$$P(B) = \frac{1}{4}(1-\theta)$$

$$P(C) = \frac{1}{4}(1-\theta)$$

$$P(D) = \frac{\theta}{4}$$

$$\theta \in (0,1)$$

Now we randomly choose one animal 197 times and get the result

$$N(A) = 125$$

$$N(B) = 18$$

$$N(C) = 20$$

$$N(D) = 34$$

```
x1=125
x2=18
x3=20
x4=34
```

2.1 MLE

We let $X = (x_1, x_2, x_3, x_4) = (125, 18, 20, 34)$

$$L(x_i;\theta) = (\frac{1}{2} + \frac{\theta}{4})^{x_1}[\frac{1}{4}(1-\theta)]^{x_2}[\frac{1}{4}(1-\theta)]^{x_3}(\frac{\theta}{4})^{x_4}$$
$$\propto (2+\theta)^{x_1}(1-\theta)^{x_2+x_3}(\theta)^{x_4}$$

$$
\begin{aligned}
l(x_i;\theta) &= \ln[(2+\theta)^{x_1}(1-\theta)^{x_2+x_3}(\theta)^{x_4}] \\
&= x_1\ln(2+\theta)+(x_2+x_3)\ln(1-\theta)+x_4\ln\theta \\
&= 125\ln(2+\theta)+38\ln(1-\theta)+34\ln\theta
\end{aligned}
$$

$$l'(x_i;\theta)=\frac{125}{2+\theta}-\frac{38}{1-\theta}+\frac{34}{\theta}=0$$

We use Newton's Method to find the result

```
f=function(x){
  125/(2+x)-38/(1-x)+34/x
}
f1=function(x){
  -125/(2+x)^2-38/(1-x)^2-34/x^2
}
x=0.1
count=0
  while (abs(f(x))>0.001&&count<1000) {
    temp=x
    x=temp-f(temp)/f1(temp)
    count=count+1
  }
x
```

```
## [1] 0.6268216
```

We get $\theta = 0.6268216$

2.2 EM

Let's suppose that we have two subspecies of A including A_1,A_2 and

$$P(A_1)=\frac{1}{2}$$

$$P(A_2)=\frac{\theta}{4}$$

Further more, suppose that

$$N(A_1)=Y$$

$$N(A_2)=x_1-Y=125-Y$$

In this situation, if we want to use MLE to get θ,we will have

$$
\begin{aligned}
L(X,Y;\theta) &= (\frac{1}{2})^Y(\frac{\theta}{4})^{x_1-Y}[\frac{1}{4}(1-\theta)]^{x_2}[\frac{1}{4}(1-\theta)]^{x_3}(\frac{\theta}{4})^{x_4} \\
&\propto (1-\theta)^{x_2+x_3}(\theta)^{x_1-Y+x_4}
\end{aligned}
$$

We can not calculate θ through this formula. Then we need EM algorithm

$$
\begin{aligned}
Q(\theta|\theta^{(i)}) &= E[\ln L(X,Y;\theta)|X,\theta^{(i)}] \\
&= E[(x_1-Y+x_4)\ln\theta+(x_2+x_3)\ln(1-\theta)|X,\theta^{(i)}] \\
&= (x_1+x_4)\ln\theta+(x_2+x_3)\ln(1-\theta)-E(Y|X,\theta^{(i)})\ln\theta
\end{aligned}
$$

When given X and $\theta^{(i)}$, $(Y|X,\theta^{(i)})\sim b(x_1,p_1)$,where $p_1=\frac{\frac{1}{2}}{\frac{1}{2}+\frac{\theta^{(i)}}{4}}=\frac{2}{2+\theta^{(i)}}$

$$E(Y|X,\theta^{(i)})=np=x_1p_1=\frac{2x_1}{2+\theta^{(i)}}$$

So

$$
\begin{aligned}
Q(\theta|\theta^{(i)}) &= (x_1+x_4)\ln\theta + (x_2+x_3)\ln(1-\theta) - E(Y|X,\theta^{(i)})\ln\theta \\
&= (x_1+x_4-\frac{2x_1}{2+\theta^{(i)}})\ln\theta + (x_2+x_3)\ln(1-\theta) \\
&= (159-\frac{250}{2+\theta^{(i)}})\ln\theta + 38\ln(1-\theta)
\end{aligned}
$$

Then we calculate the maximum of $Q(\theta|\theta^{(i)})$

$$
\begin{aligned}
Q'(\theta|\theta^{(i)}) &= (159-\frac{250}{2+\theta^{(i)}})\frac{1}{\theta} - 38\frac{1}{(1-\theta)} = 0 \\
\Rightarrow \theta^{(i+1)} &= \frac{68+159\theta^{(i)}}{144+197\theta^{(i)}}
\end{aligned}
$$

```r
EM=function(theta.init,tol,count.max){
  theta=theta.init
  count=0
  for(i in 1:count.max){
    count=count+1
    theta[i+1]=(68+159*theta[i])/(144+197*theta[i])
  if(abs(theta[i+1]-theta[i])<tol&&count<count.max)break
  }
  theta
}
theta=EM(0.2,1e-5,1000)
count=length(theta)-1
theta[length(theta)]
```

```
## [1] 0.626821
```

```r
count
```

```
## [1] 7
```

2.3 MCEM

Sometimes it is hard to find pdf, so we can also use Monte Carlo Method, which is called MCEM.

$$
\begin{aligned}
Q^{(i+1)}(\theta|\theta^{(i)}) &= \frac{1}{m}\sum_{j=1}^{m}\ln L(X,Y;\theta) \\
&= \frac{1}{m}\sum_{j=1}^{m}\ln(\theta^{x_1+x_2-Y}(1-\theta)^{x_2+x_3}) \\
&= \frac{1}{m}\sum_{j=1}^{m}(x_1+x_2-Y)\ln\theta + (x_2+x_3)\ln(1-\theta) \\
&= (x_1+x_2-E(Y))\ln\theta + (x_2+x_3)\ln(1-\theta)
\end{aligned}
$$

$$
\begin{aligned}
&Q'^{(i+1)}(\theta|\theta^{(i)}) = \frac{(x_1+x_2-E(Y))}{\theta} - \frac{(x_2+x_3)}{1-\theta} = 0 \\
&\Rightarrow \theta^{(i+1)} = \frac{x_1+x_4-E(Y)}{x_1+x_2+x_3+x_4-E(Y)} \\
&Y \sim b(x_1, \frac{2}{2+\theta^{(i)}})
\end{aligned}
$$

```
MCEM=function(theta.init,m,tol,count.max){
  theta=theta.init
  count=0
  for(i in 1:count.max){
    count=count+1
    p1=2/(2+theta)
    Y=rbinom(m,x1,p1)
    theta[i+1]=(x1+x4-mean(Y))/(x1+x2+x3+x4-mean(Y))
    if(abs(theta[i+1]-theta[i])<tol&&count<count.max)break
  }
  theta
}
theta=MCEM(0.2,10000,1e-5,1000)
count=length(theta)-1
theta[length(theta)]
```

```
## [1] 0.6246068
```

```
count
```

```
## [1] 66
```

2.4 EMNR

Sometimes the convergence speed of EM algorithm is not fast. Then we have to accelerate it. Here's one acceleration of EM algorithm. From above we have

$$\begin{aligned} Q(\theta|\theta^{(i)}) &= (x_1+x_4)\ln\theta + (x_2+x_3)\ln(1-\theta) - E(Y|X,\theta^{(i)})\ln\theta \\ &= (x_1+x_4-\frac{2x_1}{2+\theta^{(i)}})\ln\theta + (x_2+x_3)\ln(1-\theta) \\ &= (159-\frac{250}{2+\theta^{(i)}})\ln\theta + 38\ln(1-\theta) \end{aligned}$$

$$g(\theta^{(i)}) = \frac{\partial Q(\theta|\theta^{(i)})}{\partial\theta}|_{\theta=\theta^{(i)}} = (159-\frac{250}{2+\theta^{(i)}})\frac{1}{\theta^{(i)}} - \frac{38}{1-\theta^{(i)}}$$

We only need to get θ^* so that $g(\theta^*)=0$. We use Newton-Raphson to do it.

```
EMNR=function(theta.init,tol,count.max){
  theta=theta.init
  count=0
  for(i in 1:count.max){
    count=count+1
    theta[i+1]=theta[i]-((159-250/(2+theta[i]))/theta[i]-38/(1-theta[i]))/(-(159-250/(2+theta[i]))/theta
  if(abs(theta[i+1]-theta[i])<tol&&count<count.max)break
  }
  theta
}
theta=EMNR(0.2,1e-5,1000)
count=length(theta)-1
theta[length(theta)]
```

```
## [1] 0.6268215
```

```
count
```

```
## [1] 5
```

2.5 SQUAREM

Here's another way to accelerate EM algorithom.

```
loglik=function(theta){
  y=rbinom(1,x1,2/(2+theta))
  log((1-theta)^(x2+x3)*theta^(x1+x4-y))
  }
em=function(theta){
  theta.new=(68+159*theta)/(144+197*theta)
  theta=theta.new
  return(theta)
}
library(SQUAREM)
theta0=0.2
result=squarem(c(theta=theta0), fixptfn=em, objfn=loglik, control=list(tol=1e-5,mstep=5))
theta=result$par[[1]]
count=result$iter
theta
```

```
## [1] 0.6268218
```

```
count
```

```
## [1] 3
```

3 Compare

Let's choose a smaller tol to see the speed of the algorithms we talk above.

```
tol=1e-7
count.max=1000
theta.init=0.2
```

3.1 EM

```
t1=Sys.time()
theta_EM=EM(theta.init,tol,count.max)
t2=Sys.time()
iter_EM=length(theta_EM)-1
```

3.2 MCEM

```
t3=Sys.time()
theta_MCEM=MCEM(theta.init,10000,tol,count.max)
t4=Sys.time()
iter_MCEM=length(theta_MCEM)-1
```

3.3 EMNR

```
t5=Sys.time()
theta_EMNR=EMNR(theta.init,tol,count.max)
t6=Sys.time()
iter_EMNR=length(theta_EMNR)-1
```

3.4 SQUAREM

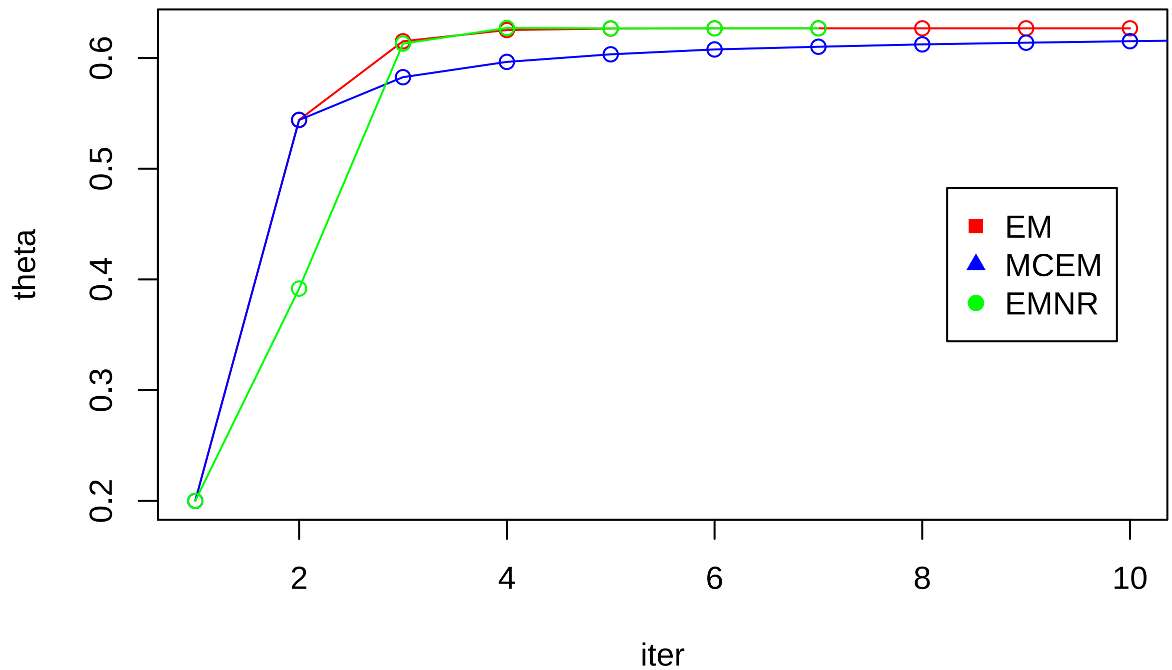
```
t7=Sys.time()
result=squarem(c(theta=theta0), fixptfn=em, objfn=loglik, control=list(tol=1e-10,mstep=5))
t8=Sys.time()
theta_SQUAREM=result$par[[1]]
iter_SQUAREM=result$iter
```

4 Result

```
table=rbind(c(theta_EM[length(theta_EM)],theta_MCEM[length(theta_MCEM)],theta_EMNR[length(theta_EMNR)],t
colnames(table)=c("EM","MCEM","EMNR","SQUAREM")
rownames(table)=c("theta","iter","run time")
table
```

```
##                 EM         MCEM      EMNR   SQUAREM
## theta    0.6268215    0.6268246 0.6268215 0.6268215
## iter     9.0000000 1000.0000000 6.0000000 5.0000000
## run time 0.0000000    1.1512301 0.0000000 0.0000000
```

```
plot(theta_EM,col="red",ylab="theta",xlab="iter")
lines(theta_EM,col="red")
points(theta_MCEM,col="blue")
lines(theta_MCEM,col="blue")
points(theta_EMNR,col="green")
lines(theta_EMNR,col="green")
legend("right", inset = .05,
       legend = c("EM","MCEM","EMNR"),
       pch = c(15,17,19),
       col = c("red","blue","green")
)
```



From the result we can see that SQUAREM is the best method because it has the fewest iterations. Newton-Raphson does accelerate EM algorithom for its fewer iterations.