EM Algorithm

Computational aspect of MLE

B. Banerjee

RTSM

"'

EM algorithm

When to use EM algorithm?

- When a data set has two parts $(\mathbf{x}, \mathbf{z})$ where $\mathbf{x}$ are observed but $\mathbf{z}$ are not observed then the maximization techniques discussed above do no lead to the MLE. Such a situation may arise in

1. Mixture distributions,
2. Hidden Markov model,
3. Incomplete data etc..

- $\ell(\theta, \mathbf{x}, \mathbf{z})$: The complete likelihood when $\mathbf{z}$ are known
- $\ell(\theta, \mathbf{x}) = \int_z \ell(\theta, \mathbf{x}, \mathbf{z})$: Marginal likelihood of $\mathbf{x}$.

EM algorithm

The EM algorithm seeks to find the MLE of the marginal likelihood by iteratively applying these two steps:

- E-STEP(Expectation step): Define $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)})$ as the expected value of the log likelihood function of $\boldsymbol{\theta}$, with respect to the current conditional distribution of $\mathbf{Z}$ given $X = \mathbf{x}$ and the current estimates of the parameters $\boldsymbol{\theta}^{(t)}$:

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}) = \mathrm{E}_{\mathbf{Z}|\mathbf{X},\boldsymbol{\theta}^{(t)}}\left[\log \ell(\boldsymbol{\theta}; \mathbf{X}, \mathbf{Z})\right]$$

,

- M-STEP(Maximization step): Find the parameters that maximize this quantity:

$$\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \ Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)})$$

,

How does the EM work?

Definition

Let f be a real valued function defined on an interval $I = [a, b]$. f is said to be convex on I if $\forall x_1, x_2 \in I$ and $\lambda \in [0, 1]$,

$$f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2).$$

Theorem

Jensen's Inequality: Let f be a convex function, and let X be a random variable. Then

$$E[f(X)] \geq f(EX).$$

- **NOTE** : If $f''(x) > 0$ for all x, then f is strictly convex

How does the EM work?

- The EM algorithm is an iterative procedure for maximizing

$$L(\theta) = \log \ell(\theta|\mathbf{x}) = \log \int_z \ell(\theta, \mathbf{x}, \mathbf{z})$$

- Assume that after the t-th iteration the current estimate for θ is given by $\theta^{(t)}$.
- Since the objective is to maximize $L(\theta) = \log \ell(\theta|\mathbf{x})$, equivalently we want to maximize the difference $L(\theta) - L(\theta^{(t)})$

$$\begin{aligned}
& L(\theta) - L(\theta^{(t)}) && (1) \\
= \;& \log \int_z \ell(\theta, \mathbf{x}, \mathbf{z}) - L(\theta^{(t)}) && (2) \\
= \;& \log \int_z \ell(\theta, \mathbf{x}|\mathbf{z}) \ell(\theta, \mathbf{z}) \frac{\ell(\theta^{(t)}, \mathbf{z}|\mathbf{x})}{\ell(\theta^{(t)}, \mathbf{z}|\mathbf{x})} - L(\theta^{(t)}) && (3) \\
\geq \;& \int_z \ell(\theta^{(t)}, \mathbf{z}|\mathbf{x}) \log \frac{\ell(\theta, \mathbf{x}|\mathbf{z}) \ell(\theta, \mathbf{z})}{\ell(\theta^{(t)}, \mathbf{z}|\mathbf{x}) \ell(\theta^{(t)}, \mathbf{x})} = \Delta(\theta||\theta^{(t)}) && (4)
\end{aligned}$$

How does the EM work?

- $L(\theta) \geq L(\theta^{(t)}) + \Delta(\theta||\theta^{(t)})$ where $\Delta(\theta||\theta^{(t)}) = 0$ if $\theta = \theta^{(t)}$
- E-STEP(Expectation step): The current estimates of the parameters $\boldsymbol{\theta}^{(t)}$:

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}) = \mathrm{E}_{\mathbf{Z}|\mathbf{X},\boldsymbol{\theta}^{(t)}}\left[\log \ell(\boldsymbol{\theta};\mathbf{X},\mathbf{Z})\right]$$

,

- M-STEP(Maximization step):

$$\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \ Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)})$$

,

How does the EM work?

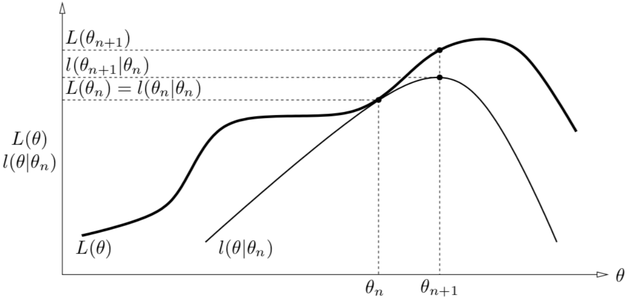


Figure 2: Graphical interpretation of a single iteration of the EM algorithm: The function $L(\theta|\theta_n)$ is upper-bounded by the likelihood function $L(\theta)$. The functions are equal at $\theta = \theta_n$. The EM algorithm chooses θ_{n+1} as the value of θ for which $l(\theta|\theta_n)$ is a maximum. Since $L(\theta) \geq l(\theta|\theta_n)$ increasing $l(\theta|\theta_n)$ ensures that the value of the likelihood function $L(\theta)$ is increased at each step.

Kullback-Leibler divergence

- The Kullback-Leibler divergence was introduced by Solomon Kullback and Richard Leibler in 1951 as the directed divergence between two distributions.
- For discrete valued random variables in X

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_x P(x) \log\left(\frac{P(x)}{Q(x)}\right).$$

For continuous valued random variable

$$D_{\mathrm{KL}}(P \parallel Q) = \int_{-\infty}^{\infty} p(x) \log\left(\frac{p(x)}{q(x)}\right)\, dx$$

- KL divergence is NOT a distance measure or metric because it is not symmetric.

-The Kullback-Leibler divergence is always non-negative i.e.

$$D_{\mathrm{KL}}(P \parallel Q) \geq 0$$

EM: Example

- Suppose $Y = (y_1, y_2, y_3, y_4)$ has a multinomial distribution with cell probabilities

$$\left(\frac{1}{2} + \frac{\theta}{4}, \frac{1-\theta}{4}, \frac{1-\theta}{4}, \frac{\theta}{4}\right)$$

  Find the MLE of θ when observed $Y = (125, 18, 20, 34)$

- Solution: Define the complete-data: $X = (x_0, x_1, y_2, y_3, y_4)$ to have a multinomial distribution with probabilities

$$\left(\frac{1}{2}, \frac{\theta}{4}, \frac{1-\theta}{4}, \frac{1-\theta}{4}, \frac{\theta}{4}\right)$$

  shuch that $y_1 = x_0 + x_1$

- Observed-data log likelihood

$$l(\theta|Y) = y_1 \log(0.5 + \theta/2) + (y_2 + y_3) \log(1 - \theta) + y_4 \log \theta$$

- Complete-data log likelihood

$$l_c(\theta|X) = (x_1 + y_4) \log \theta + (y_2 + y_3) \log(1 - \theta)$$

- E step:

$$x^{(n+1)} = E(x_1|Y, \theta^{(n)}) = y_1 \frac{\theta^{(n)}/4}{0.5 + \theta^{(n)}/4}$$

- M step:

$$\theta^{(n+1)} = \frac{x_1^{(n+1)} + y_4}{x_1^{(n+1)} + y_2 + y_3 + y_4}$$

EM: Example

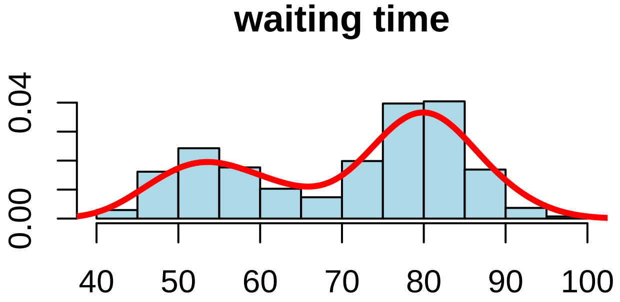
Steps	$\theta^{(n)}$
0	0.500000000
1	0.608247423
2	0.624321051
3	0.626488879
4	0.626777323
5	0.626815632
6	0.626820719
7	0.626821395
8	0.626821484
9	0.626821498

EM : Example

- Waiting time between eruptions and the duration of the eruption for the Old Faithful geyser in Yellowstone National Park, Wyoming, USA.A data frame with 272 observations on 2 variables.
- eruptions: Eruption time in mins
- waiting: Waiting time to next eruption

EM : Example

```r
hist(faithful$waiting,
     probability=TRUE, breaks=10, col="light blue",
     xlab="", ylab="",
     main="waiting time")
lines(density(faithful$waiting), type='l',
       col='red', lwd=3)
```



EM : Example

- X: Waiting time
- p: probability of shorter waiting time
- $\boldsymbol{\theta} = (p, \mu_1, \sigma_1^2, \mu_2, \sigma_2^2)$
- $f(x|\boldsymbol{\theta}) = p\frac{1}{\sigma_1}\phi(\frac{x-\mu_1}{\sigma_1}) + (1-p)\frac{1}{\sigma_2}\phi(\frac{x-\mu_2}{\sigma_2})$
- $Z_i \sim Bernoulli(p)$ which are missing/unobserved/ latent variable
- $Z_i = 1$ if shorter waiting time and 0 otherwise.

EM : Example

- **E-step:** $Z_i|X_i, \boldsymbol{\theta}^{(k)} \sim Bernoulli(p_i^{(k)})$. So,

$$E(Z_i|X_i, \boldsymbol{\theta}^{(k)}) = \frac{p^{(k)} \frac{1}{\sigma_1^{(k)}} \phi\left(\frac{x - \mu_1^{(k)}}{\sigma_1^{(k)}}\right)}{f(x_i, \boldsymbol{\theta}^{(k)})}$$

- **M-step:** Complete data likelihood

$$l(\boldsymbol{\theta}|\mathbf{x}, \mathbf{z}) = \prod_i \left[p \frac{1}{\sigma_1} \phi\left(\frac{x - \mu_1}{\sigma_1}\right)\right]^{z_i} \left[(1-p) \frac{1}{\sigma_2} \phi\left(\frac{x - \mu_2}{\sigma_2}\right)\right]^{1-z_i}$$

replace z_i by $p_i^{(k)}$ and maximize for $\boldsymbol{\theta}$ to get the following

EM : Example

- $p^{(k+1)} = \frac{1}{n} \sum_i p_i^{(k)}$
- $\mu_1^{(k+1)} = \frac{\sum_i p_i^{(k)} X_i}{\sum_i p_i^{(k)}}$
- $\sigma_1^{2^{(k+1)}} = \frac{\sum_i p_i^{(k)} (X_i - \mu_1^{(k+1)})^2}{\sum_i p_i^{(k)}}$
- $\mu_2^{(k+1)} = \frac{\sum_i (1 - p_i^{(k)}) X_i}{\sum_i (1 - p_i^{(k)})}$
- $\sigma_2^{2^{(k+1)}} = \frac{\sum_i (1 - p_i^{(k)})(X_i - \mu_2^{(k+1)})^2}{\sum_i (1 - p_i^{(k)})}$

EM : Example

EM Algorithm

B. Banerjee

EM algorithm

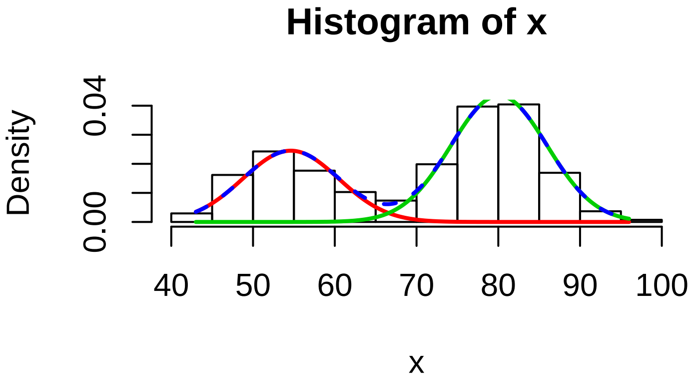
```
emfun<-function(x,th){
  Ep<-th[1]*dnorm(x,th[2],sqrt(th[3]))/(th[1]*dnorm(x,th[2],sqrt(th[3]))+(1-th[1])*dnorm(x,th[4],sqrt(th[5])))
  th[1]<-mean(Ep)
  th[2]<-sum(Ep*x)/sum(Ep)
  th[3]<-sum(Ep*(x-th[2])^2)/sum(Ep)
  th[4]<-sum((1-Ep)*x)/sum(1-Ep)
  th[5]<-sum((1-Ep)*(x-th[4])^2)/sum(1-Ep)

 th
}

x<-faithful$waiting
y<-seq(min(x),max(x),by=0.2)
fd<-which(x<mean(range(x)))
th<-c(length(fd)/length(x), mean(x[fd]),var(x[fd]), mean(x[-fd]),var(x[-fd]))
cat(0,"iteratin:",th,"\n")
hist(x, probability = T)
s1<-emfun(x,th)
ct=1
cat(ct,"iteratin:",s1,"\n")
cutoff<-rep(0.001,5)
while(sum((th-s1)>cutoff)>0){
  th<-s1
  hist(x, probability = T)
  s1<-emfun(x,th)
  ct=ct+1
}
cat(ct,"iteratin:",s1,"\n")
lines(th[1]*dnorm(y,th[2],sqrt(th[3]))~y,col=2, lwd=2)
lines((1-th[1])*dnorm(y,th[4],sqrt(th[5]))~y,col=3, lwd=2)
lines(th[1]*dnorm(y,th[2],sqrt(th[3]))+(1-th[1])*dnorm(y,th[4],sqrt(th[5]))~y,col=4, lwd=2,lty=2)
```

EM : Example

```
## 0 iteratin: 0.3786765 55.15534 39.26975 80.49112 29.77522
```

```
## 1 iteratin: 0.3720185 54.99768 38.53527 80.31591 31.93419
```

```
## 20 iteratin: 0.3608899 54.61498 34.4725 80.09115 34.42936
```



EM : Hands-on 1


```r
emfun<-function(x,th){
  Ep<-th[1]*dnorm(x,th[2],sqrt(th[3]))/(th[1]*dnorm(x,th[2],sqrt(th[3]))+(1-th[1])*dnorm(x,th[4],sqrt(th[5])))
  th[1]<-mean(Ep)
  th[2]<-sum(Ep*x)/sum(Ep)
  th[3]<-sum(Ep*(x-th[2])^2)/sum(Ep)
  th[4]<-sum((1-Ep)*x)/sum(1-Ep)
  th[5]<-sum((1-Ep)*(x-th[4])^2)/sum(1-Ep)
 th
}
x<- c(rnorm(100,1,1), rnorm(200,5,1.5)) # enable 1
#x<-faithful$waiting # data   # disable 1
y<-seq(min(x),max(x),by=0.2)
fd<-which(x<mean(range(x)))
th<-c(length(fd)/length(x), mean(x[fd]),var(x[fd]), mean(x[-fd]),var(x[-fd]))
hist(x, probability = T)
s1<-emfun(x,th)
ct=1
cat(ct,s1,"\n")
cutoff<-rep(0.001,5)
while(sum((th-s1)>cutoff)>0){

  th<-s1
  hist(x, probability = T)
 lines(th[1]*dnorm(y,th[2],sqrt(th[3]))~y,col=2, lwd=2) # enable 2
  lines((1-th[1])*dnorm(y,th[4],sqrt(th[5]))~y,col=3, lwd=2) # enavle 3
  s1<-emfun(x,th)
  ct=ct+1
  cat(ct,"iteratin:",s1,"\n")
  Sys.sleep(0.5)
}
lines(th[1]*dnorm(y,th[2],sqrt(th[3]))~y,col=2, lwd=2)
lines((1-th[1])*dnorm(y,th[4],sqrt(th[5]))~y,col=3, lwd=2)
lines(th[1]*dnorm(y,th[2],sqrt(th[3]))+(1-th[1])*dnorm(y,th[4],sqrt(th[5]))~y,col=4, lwd=2,lty=3)
```

EM : Example (2D)

EM Algorithm

B. Banerjee

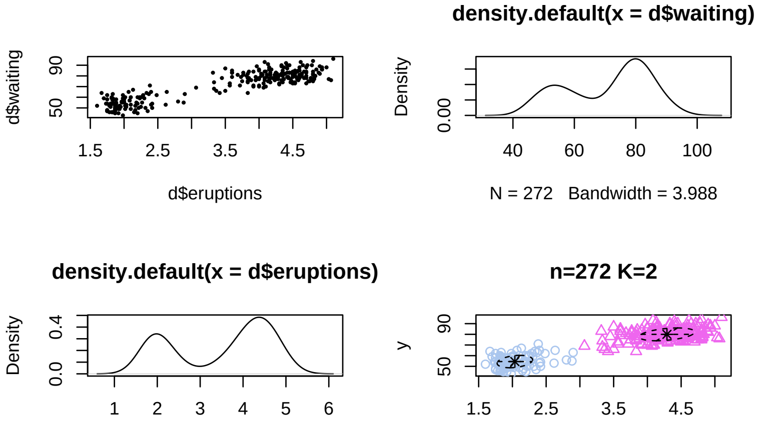
EM algorithm

```r
d<-faithful[1:2]
par(mfrow=c(2,2))
plot(d$waiting~d$eruptions, pch = 20, cex = 0.5, col=1)
plot (density (d$waiting))
plot (density (d$eruptions))
library("EMCluster", quietly = F)
k<- 2 # number of  clusters
p<- 2 # dimention
#emobj <- simple.init(d, nclass = k)
#emobj <- shortemcluster(d, emobj)
mm<- array(0,dim=c(k,p))
mm[,1]<- c(quantile(d$eruptions,probs = 0.25), quantile(d$eruptions,probs = 0.75))
mm[,2]<- c(quantile(d$waiting,probs = 0.25), quantile(d$waiting,probs = 0.75))
covm<-array(0,dim=c(k,p*(p+1)/2))
d1<-d[which(d$eruptions<3),]
d2<-d[which(d$eruptions>3),]
covm[1,]<-c(var(d1[,1]),cov(d1[,1],d1[,2]),var(d1[,2]))
covm[2,]<-c(var(d2[,1]),cov(d2[,1],d2[,2]),var(d2[,2]))
ret <- emcluster(d, pi = c(0.5,0.5),Mu = mm, LTSigma = covm, assign.class = T )
#print(summary(ret))
plotem(ret,d)
```

https://www.mathworks.com/matlabcentral/fileexchange/49869-expectation-maximization-on-old-faithful

EM : Example (2D)

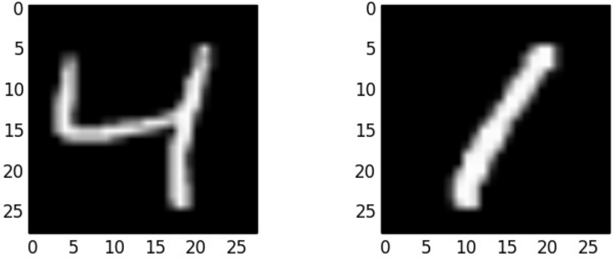
```
## Warning: package 'EMCluster' was built under R version 3.4.4

## Loading required package: MASS

## Loading required package: Matrix
```



EM : Hands-on 2 : Unsuperviser Digits 1 & 4


```r
dig<-read.csv("test.csv")
da<-which(dig[,1]==1)
db<-which(dig[,1]==4)
dd<-rbind(dig[da,-1],dig[db,-1])
rcdd<-dim(dd)
set.seed(890)
noise<-runif( prod(rcdd),0,.001)
noisem<-matrix(data = noise,nrow = rcdd[1],byrow = T)
dd<-dd+noisem
prin_comp<-prcomp(dd,center = F,scale. = T)
pcvar <- (prin_comp$sdev)^2
plot(cumsum(pcvar)/sum(pcvar), type=l)
ddpc<-prin_comp$x[,1:14]
library("EMCluster", quietly = F)
k<- 2 # number of  clusters
p<- ncol(ddpc) # dimention
# The simple.init utilizes rand.EM to obtain a simple initial.
emobj <- simple.init(ddpc, nclass = 2)
#The best of several random initializations.
emobj <- shortemcluster(ddpc, emobj)
ret <- emcluster(ddpc, emobj, assign.class = T )
# print(ret$Mu)
# print(ret$LTSigma)
cat("number of iteration=", ret$conv.iter,"\n")
cat("estimated  proportion=",ret$pi,"\n")
cat("True proportion=", length(da)/(length(da)+length(db)),length(db)/(length(da)+length(db)),"\n")
crossv<-matrix(c(sum(which(ret$class==1)<length(da))/nrow(dd),
c(sum(which(ret$class==1)>length(da))/nrow(dd)),
sum(which(ret$class==2)<length(da))/nrow(dd),
sum(which(ret$class==2)>length(da))/nrow(dd)),nrow = 2,byrow = T)
cat("Diagonal entries stand for correct identification", "\n")
print(crossv)
```

EM : Hands-on 2 : Unsuperviser Digits 1 & 4



EM : Hands-on 2 : Unsuperviser Digits 1 & 4

```
## number of iteration= 6

## estimated  proportion= 0.5955723 0.4044277

## True proportion= 0.536136 0.463864

## Diagonal entries stand for correct identification

##            [,1]       [,2]
## [1,] 0.51393481 0.08219178
## [2,] 0.02172886 0.38167218
```