#第一題

theta\_0<- 0.5

theta<- 0

theta\_2<- theta\_0

n<- 0 # by increasing sample size to control monte carlo estimation

Theta<-c() # storge theta

g<-function(x){ x\*factorial(125)/factorial(x)/factorial(125-x) \*( theta/(2+theta) )^x \*( 2/(2+theta) )^(125-x) }

Q<-function(x){ 38\*log(1/2-x/2) +34\*log(x/4) +log(x/4)\*I\_y3 }

while(abs(theta-theta\_2) >= 10^-7){

n<-n+1000

cat("now theta is",theta\_2,"\n")

theta <-theta\_2

# estimate E(Y3|X)

I\_y3<-sum( g(runif(n,0,125)) ) /n \*125

# max Q(theta)

h<-10^-6

x1<-0 ;x2<-0.5

while(abs(x2-x1)>= 10^-5 ){

x1<-x2

firdif<-(Q(x1+h)-Q(x1))/h

secdif<-(Q(x1+2\*h)-2\*Q(x1+h)+Q(x1) ) / h^2

x2<-x1 -firdif/secdif

}

theta\_2<-x2

Theta<-c(Theta,theta\_2)

}

# observed data (Y1,Y2) ~ multnomial(72 ,p1' ,p2')

log.obser<-function(x){ 38\*log( (2-2\*x)/(2-x) ) +34\*log( x/(2-x) ) }

plot(log.obser(Theta), type = "l" ,main = "MC-step" ,xlab = "i-th iteration")

#第二題

# simulation of mixture normal

# true par

p<-c(p1= 0.2 ,p2= 0.3 ,p3= 0.5)

mu<-c(mu1= -3, mu2= 0, mu3= 2)

sigma.sqr<-c(sigma1.sqr= 0.64, sigma2.sqr= 0.36, sigma3.sqr= 0.25)

## generate data :2000 (400, 1400 ,200)

G<-function(z,n){

z<-c(z,rep(0,n))

for(i in 1:n){z[i+1]<-(16807\*z[i])%%(2^31-1)}

u<-z[-1]/(2^31-1) }

#acceptance rejection of N(0,1) with g(x)~exp(1)

windows();plot(x=c(seq(0,3,0.001),seq(0,3,0.001)),y=c(dnorm(seq(0,3,0.001),0,1),2\*dexp(seq(0,3,0.001),1)),

col =c(rep(1,3001),rep(2,3001)) )

envo<-function(x,c){ (sqrt(2/pi)\*exp(-x^2/2)) / (c\*exp(-x)) }

acc.rej.exp<-function(n,c){

u1<-G(1,c\*n)

Y<--log(u1)

u2<-G(u1[c\*n],c\*n)

X<-rep(0,n)

N<-length(which(u2 <= envo(Y,c) ))

for(i in 1:min(n,N) ){ X[i]<-Y[which( u2 <= envo(Y,c) )][i] } #exp(-(Y-1)^2/2)

uu2<-rep(0,n)

uu2[n]<-u2[c\*n]

while(N<n){

uu1<-G(uu2[n],n)

Y<--log(uu1)

uu2<-G(uu1[n],n)

if(length(which(uu2<=envo(Y,c)))>0){

for(i in 1:min(n,length(which(uu2<=envo(Y,c) ) )) ){ X[N+i]<-Y[which(uu2<=envo(Y,c))][i] } }

N<-N+length(which(uu2 <= envo(Y,c)))

}

u3<-G(uu2[n],n)

X[which(u3<=0.5)]<--X[which(u3<=0.5)]

return(X[1:n])

}

A<-acc.rej.exp(2000,2)

windows();qqnorm(A);qqline(A)

ks.test(A ,"pnorm" ,0 ,1)

# X1

X1<- sqrt(sigma.sqr[1])\*A[1:(2000\*p[1])] +mu[1]

ks.test(X1 ,"pnorm" ,mu[1] ,sqrt(sigma.sqr[1]))

# X2

X2<- sqrt(sigma.sqr[2])\*A[(2000\*p[1]+1):(2000\*(p[1]+p[2]))] +mu[2]

ks.test(X2 ,"pnorm" ,mu[2] ,sqrt(sigma.sqr[2]))

# X3

X3<- sqrt(sigma.sqr[3])\*A[(2000\*(p[1]+p[2])+1):2000] +mu[3]

ks.test(X3 ,"pnorm" ,mu[3] ,sqrt(sigma.sqr[3]))

# observed data X=(X1,X2,X3)

obs.X<-c(X1,X2,X3)

rm(A,X1,X2,X3)

hist(obs.X ,breaks = 100)

windows();qqnorm(obs.X);qqline(obs.X)

## EM algorithm

# initial value

p1\_0<- 0.2 ; mu1\_0<- -3 ; sigma1.sqr\_0<- 0.64

p2\_0<- 0.3 ; mu2\_0<- 0 ; sigma2.sqr\_0<- 0.36

p3\_0<- 0.5 ; mu3\_0<- 2 ; sigma3.sqr\_0<- 0.25

p1\_2<- p1\_0

p2\_2<- p2\_0

p3\_2<- p3\_0

mu1\_2<- mu1\_0

mu2\_2<- mu2\_0

mu3\_2<- mu3\_0

sigma1.sqr\_2<- sigma1.sqr\_0

sigma2.sqr\_2<- sigma2.sqr\_0

sigma3.sqr\_2<- sigma3.sqr\_0

p1\_1<- p1\_2

p2\_1<- p2\_2

p3\_1<- p3\_2

mu1\_1<- mu1\_2

mu2\_1<- mu2\_2

mu3\_1<- mu3\_2

sigma1.sqr\_1<- sigma1.sqr\_2

sigma2.sqr\_1<- sigma2.sqr\_2

sigma3.sqr\_1<- sigma3.sqr\_2

# latent: Z\_i1 ,Z\_i2 ,Z\_i3 update

f<-function(x,mu,sigma.sqr){ 1/sqrt(2\*pi\*sigma.sqr) \*exp( -(x-mu)^2 /sigma.sqr) }

Z\_i1<- ( p1\_1 \*f(obs.X ,mu1\_1 ,sigma1.sqr\_1) ) /( p1\_1 \*f(obs.X ,mu1\_1 ,sigma1.sqr\_1) + p2\_1 \*f(obs.X ,mu1\_1 ,sigma1.sqr\_1) + p3\_1 \*f(obs.X ,mu1\_1 ,sigma1.sqr\_1) )

Z\_i2<- ( p2\_1 \*f(obs.X ,mu2\_1 ,sigma2.sqr\_1) ) /( p1\_1 \*f(obs.X ,mu2\_1 ,sigma2.sqr\_1) + p2\_1 \*f(obs.X ,mu2\_1 ,sigma2.sqr\_1) + p3\_1 \*f(obs.X ,mu2\_1 ,sigma2.sqr\_1) )

Z\_i3<- ( p3\_1 \*f(obs.X ,mu3\_1 ,sigma3.sqr\_1) ) /( p1\_1 \*f(obs.X ,mu3\_1 ,sigma3.sqr\_1) + p2\_1 \*f(obs.X ,mu3\_1 ,sigma3.sqr\_1) + p3\_1 \*f(obs.X ,mu3\_1 ,sigma3.sqr\_1) )

Z<-cbind(Z\_i1,Z\_i2,Z\_i3)

# p1 ,p2 ,p3 update

p1\_2<- sum(Z\_i1) /sum(Z\_i1+Z\_i2+Z\_i3)

p2\_2<- sum(Z\_i2) /sum(Z\_i1+Z\_i2+Z\_i3)

p3\_2<- sum(Z\_i3) /sum(Z\_i1+Z\_i2+Z\_i3)

# mu1 ,mu2 ,mu3 update

mu1\_2<- as.vector(Z\_i1 %\*% obs.X /sum(Z\_i1))

mu2\_2<- as.vector(Z\_i2 %\*% obs.X /sum(Z\_i2))

mu3\_2<- as.vector(Z\_i3 %\*% obs.X /sum(Z\_i3))

# sigma1^2 ,sigma2^2 ,sigma3^2 update

sigma1.sqr\_2<- as.vector((obs.X - mu1\_2)^2 %\*% Z\_i1 /sum(Z\_i1))

sigma2.sqr\_2<- as.vector((obs.X - mu2\_2)^2 %\*% Z\_i2 /sum(Z\_i2))

sigma3.sqr\_2<- as.vector((obs.X - mu3\_2)^2 %\*% Z\_i3 /sum(Z\_i3))

#第三題

#############################################

#C1: x\_0 = 0 ; n = 1000 ; different epslon #

#############################################

#case: x\_0 = 0 ; epslon = 1

x\_0<- 0

epslon<- 1

x\_2<- x\_0

X<- c()

n<-1000

for(i in 1:n){

x\_1<- x\_2

# generate y ~ U( x\_1-epslon ,x\_1+epslon )

y<- runif(1, x\_1-epslon ,x\_1+epslon)

# acceptance prob.

A<-function(x,y){ min( exp(-y^2/2 +x^2/2), 1 ) }

u<- runif(1, 0, 1)

if( u <= A(x\_1,y) ){ x\_2<- y }else{ x\_2<- x\_1 }

X<- c(X,x\_2)

}

windows();par(mfrow=c(3,2))

plot(X ,type = "l" ,xlab = "n=1000" ,ylab = "X\_n" ,col = "1",main ="x\_0=0 ,epslon=1" )

qqnorm(X);qqline(X)

#case: x\_0 = 0 ; epslon = 0.2

x\_0<- 0

epslon<- 0.2

x\_2<- x\_0

X<- c()

n<-1000

for(i in 1:n){

x\_1<- x\_2

# generate y ~ U( x\_1-epslon ,x\_1+epslon )

y<- runif(1, x\_1-epslon ,x\_1+epslon)

# acceptance prob.

A<-function(x,y){ min( exp(-y^2/2 +x^2/2), 1 ) }

u<- runif(1, 0, 1)

if( u <= A(x\_1,y) ){ x\_2<- y }else{ x\_2<- x\_1 }

X<- c(X,x\_2)

}

plot(X ,type = "l" ,xlab = "n=1000" ,ylab = "X\_n" ,col = "1",main ="x\_0=0 ,epslon=0.2" )

qqnorm(X);qqline(X)

#case: x\_0 = 0 ; epslon = 20

x\_0<- 0

epslon<- 20

x\_2<- x\_0

X<- c()

n<-1000

for(i in 1:n){

x\_1<- x\_2

# generate y ~ U( x\_1-epslon ,x\_1+epslon )

y<- runif(1, x\_1-epslon ,x\_1+epslon)

# acceptance prob.

A<-function(x,y){ min( exp(-y^2/2 +x^2/2), 1 ) }

u<- runif(1, 0, 1)

if( u <= A(x\_1,y) ){ x\_2<- y }else{ x\_2<- x\_1 }

X<- c(X,x\_2)

}

plot(X ,type = "l" ,xlab = "n=1000" ,ylab = "X\_n" ,col = "1",main ="x\_0=0 ,epslon=20" )

qqnorm(X);qqline(X)

###########################################

#C2: x\_0 = -50 ; epslon = 1 ; different n #

###########################################

#case: x\_0 = -50 ; epslon = 1 ; n = 1000

x\_0<- -50

epslon<- 1

x\_2<- x\_0

X<- c()

n<-1000

for(i in 1:n){

x\_1<- x\_2

# generate y ~ U( x\_1-epslon ,x\_1+epslon )

y<- runif(1, x\_1-epslon ,x\_1+epslon)

# acceptance prob.

A<-function(x,y){ min( exp(-y^2/2 +x^2/2), 1 ) }

u<- runif(1, 0, 1)

if( u <= A(x\_1,y) ){ x\_2<- y }else{ x\_2<- x\_1 }

X<- c(X,x\_2)

}

windows();par(mfrow=c(4,2))

plot(X ,type = "l" ,xlab = "n=1000" ,ylab = "X\_n" ,col = "1",main = "x\_0=-50, epslon=1, without burn-in")

qqnorm(X);qqline(X)

plot(X[(n/2):n] ,type = "l" ,xlab = "n=1000" ,ylab = "X\_n" ,main = "x\_0=-50, epslon=1, burn-in with 500 samples" ,cex.main =1)

qqnorm(X[(n/2):n]);qqline(X[(n/2):n])

#case: x\_0 = -50 ; epslon = 10 ; n = 1000

x\_0<- -50

epslon<- 10

x\_2<- x\_0

X<- c()

n<-1000

for(i in 1:n){

x\_1<- x\_2

# generate y ~ U( x\_1-epslon ,x\_1+epslon )

y<- runif(1, x\_1-epslon ,x\_1+epslon)

# acceptance prob.

A<-function(x,y){ min( exp(-y^2/2 +x^2/2), 1 ) }

u<- runif(1, 0, 1)

if( u <= A(x\_1,y) ){ x\_2<- y }else{ x\_2<- x\_1 }

X<- c(X,x\_2)

}

plot(X ,type = "l" ,xlab = "n=1000" ,ylab = "X\_n" ,col = "1",main = "x\_0=-50, epslon=10, without burn-in")

qqnorm(X);qqline(X)

plot(X[(n/4):n] ,type = "l" ,xlab = "n=1000" ,ylab = "X\_n" ,main = "x\_0=-50, epslon=10, burn-in with 250 samples" ,cex.main =1)

qqnorm(X[(n/4):n]);qqline(X[(n/4):n])

###############################################

# to estimate E( exp(Z^16) )

#case: x\_0 = 0 ; epslon = 1

x\_0<- 0

epslon<- 1

x\_2<- x\_0

X<- c()

n<-1000

for(i in 1:n){

x\_1<- x\_2

# generate y ~ U( x\_1-epslon ,x\_1+epslon )

y<- runif(1, x\_1-epslon ,x\_1+epslon)

# acceptance prob.

A<-function(x,y){ min( exp(-y^2/2 +x^2/2), 1 ) }

u<- runif(1, 0, 1)

if( u <= A(x\_1,y) ){ x\_2<- y }else{ x\_2<- x\_1 }

X<- c(X,x\_2)

}

windows();par(mfrow=c(3,2))

plot(X ,type = "l" ,xlab = "n=1000" ,ylab = "X\_n" ,col = "1",main ="x\_0=0 ,epslon=1" )

qqnorm(X);qqline(X)

I.hat<- mean( exp(X^16) )