Statistical Computing Final Exam

610611104 Pan, Hsing-Cheng

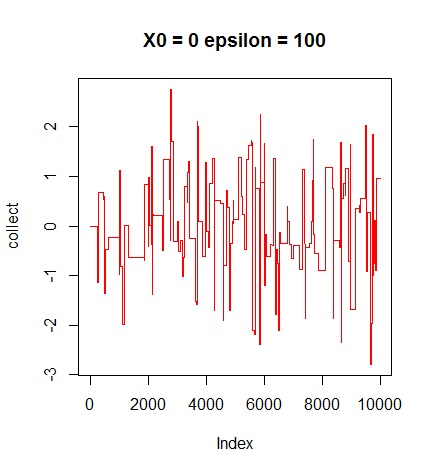
1.(i)

根據上次作業的推導基本上已推導出E-step跟M-step，這次則是在E-step的時候，利用Monte Carlo method估計的期望值。而我們知道給定()的分佈，所以我們的Monte Carlo method是利用生成125個uniform(0,1)的variable，如果uniform的值小於給定當次theta的機率則加一，重複N次取平均。

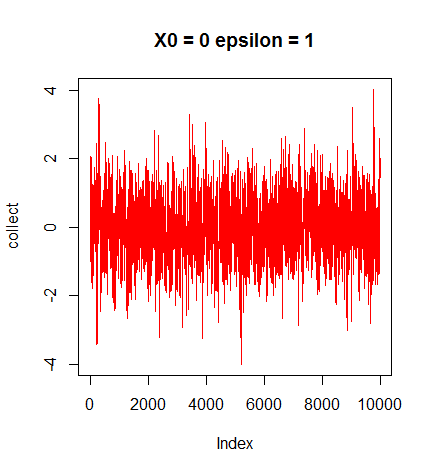
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Initial value | iteration | Converge value | error | Method |
| 0.5 | 8 | 0.6268215 |  | 1 |
| 0.5 | 40 | 0.6268257 |  | 2 |

3.(i)

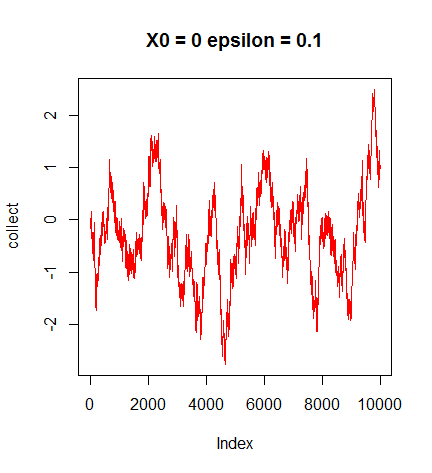
根據SamplingBurning.pdf裡的介紹利用Metropolis-Hasting method從uniform(x-epsilon,x+epsilon)生成standard normal的值並觀察他的表現。



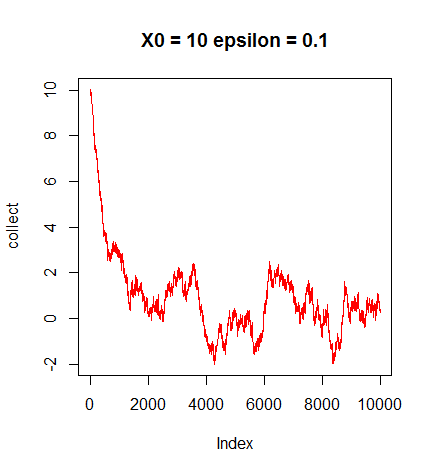
從圖中可以看到他的迭代值跳躍很大，而且並沒有很隨機的跳動的感覺，所以在這設定下是不好的。



在這個設定下，圖中的表現算是不錯的，很快就進入穩定狀態。



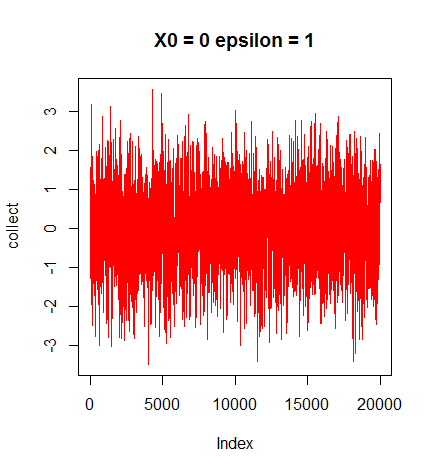
在這個設定下他的跳躍的相當緩慢，並且與時間的長度有著很強的相關性，而且經過了10000次的暖機也還沒有收斂到穩定狀態。



在這個設定下很明顯看到起始值離穩定狀態的範圍很遠，而且也需用運行長時間先當作暖機，之後生成的值才是穩定狀態的sample。

3.(ii)

利用(i)的方法並設定=0和epsilon=1的情況下，生成20000個值，並把前10000個值當作暖機的丟掉，只取後10000個值來計算。



但因為要先取16次方在取exponential，因為值太大有一些值會顯示Inf，而使得取樣本平均的時候會顯示Inf。

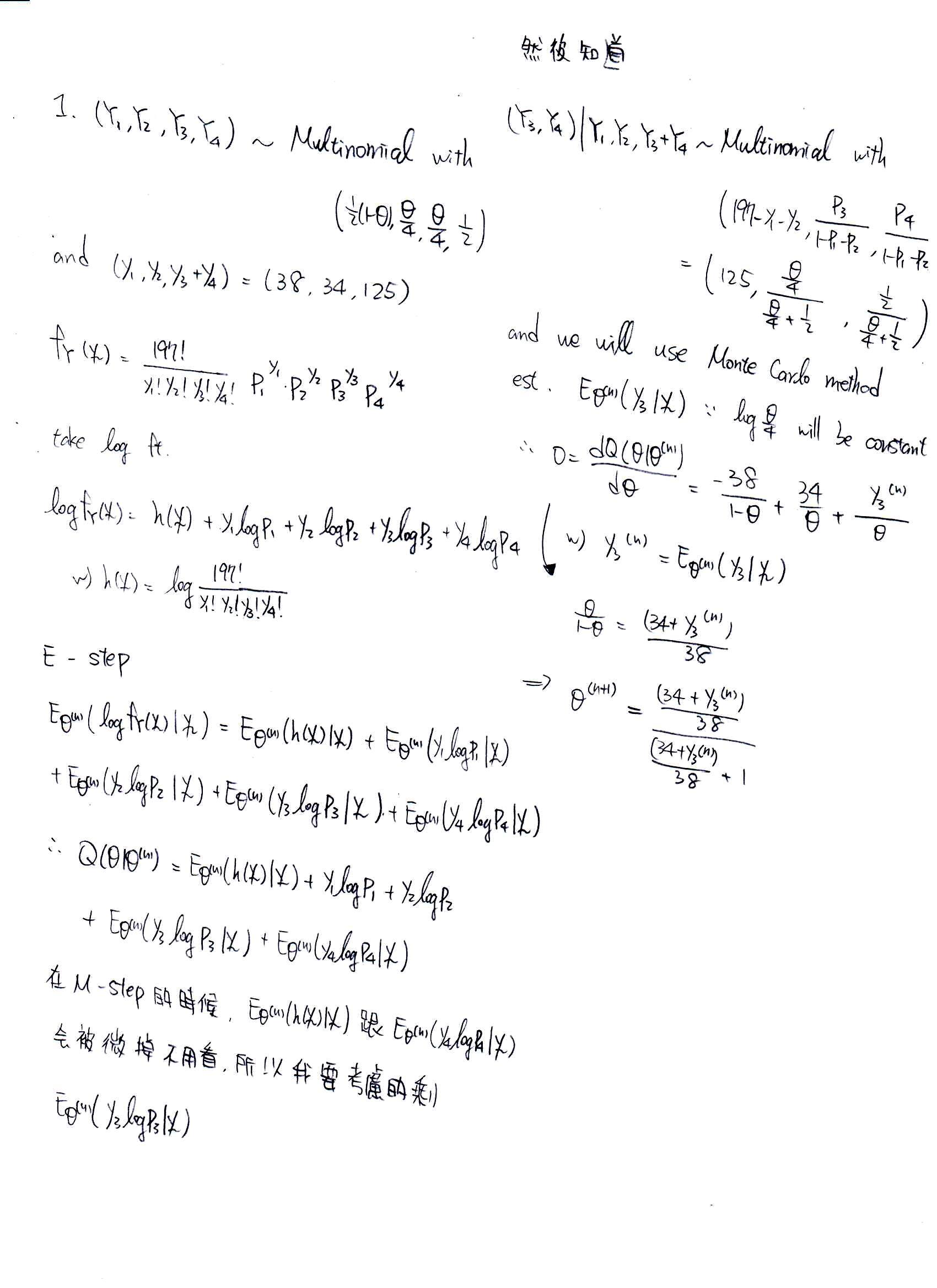
2.(b)

根據(i)推出的E-M algorithm，自己設定真實參數以及初始值，帶入推出來的演算法來檢查是否正確。

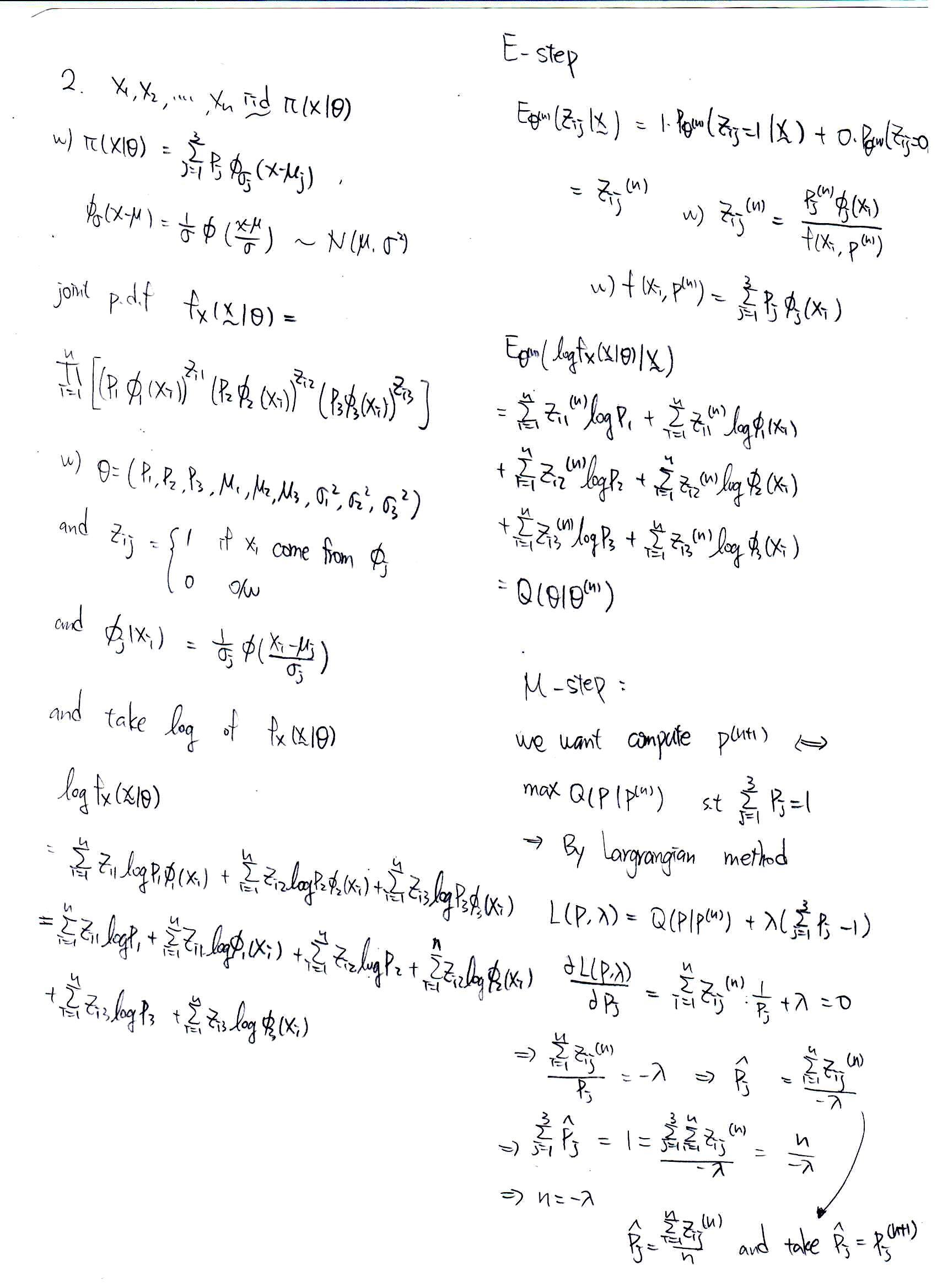
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 真實參數 | P1=0.5 | P2=0.3 | P3=0.2 | Mu1=2 | Sigma1=1 | Mu2=3 | Sigma2=9 | Mu3=0 | Sigma3=1 |
| 初始值 | P1=0.22 | P2=0.28 | P3=0.5 | Mu1=10 | Sigma1=10 | Mu2=5 | Sigma2=5 | Mu3=3 | Sigma3=1/2 |
| 收斂值 | P1=0.101 | P2=0.579 | P3=0.319 | Mu1=5.81 | Sigma1=4.31 | Mu2=1.182 | Sigma2=3.29 | Mu3=2 | Sigma3=0.93 |

在模擬中當前一次跟這次迭代參數誤差最大值小於則停止。不過不曉得什麼原因他無法很好的收斂到我設定的真實參數。(迭代了1790次)

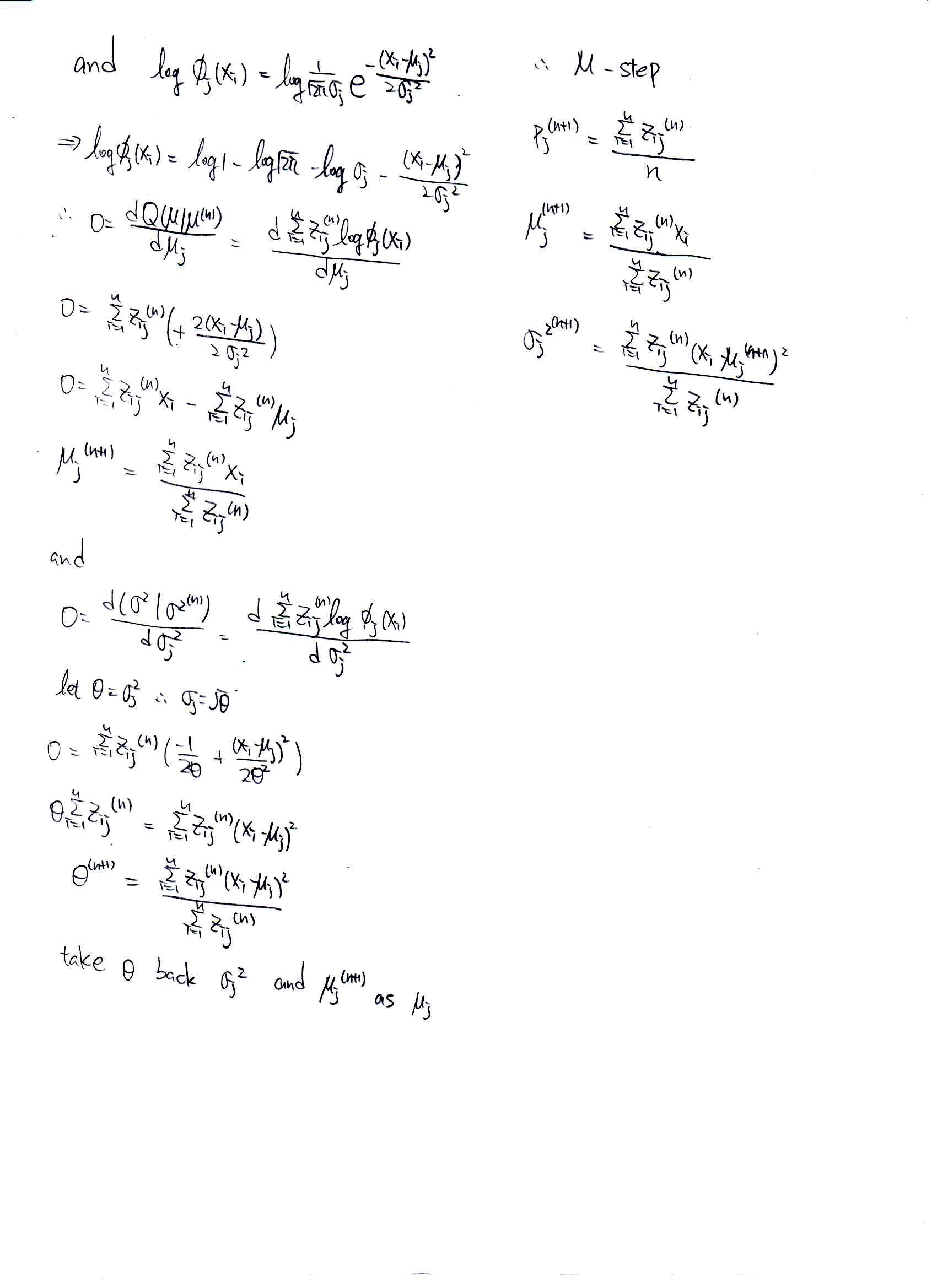
1. Writing



2.(a)-1writing



2.(a)-2 writing



Coding:

1.

|  |
| --- |
| rm(list=ls())  graphics.off()  #####  initialpar<-0.5  beforepar<-initialpar  est.y3<-function(n,theta){  y3<-numeric(n)  for(j in 1:n){  y<-0  for(i in 1:125){  u1<-runif(1,0,1)  if(u1<=theta){y<-y+1}  }  y3[j]<-y  }  return(mean(y3))  }  initial.y3t<-(beforepar/4)/(beforepar/4+1/2)  before.y3t<-initial.y3t  conv<-FALSE  loop<-1  while(!conv){  y3n<-est.y3(10000,before.y3t)  newpar<-((34+y3n)/38)/((34+y3n)/38+1)  afterpar<-newpar  cat("now is:",loop,"th loop and mean of y3 is:",y3n,"\n")  cat("now is:",loop,"th loop and (n+1)th theta is:",afterpar,"(n+1)th theta - nth theta is:",abs(afterpar-beforepar),"\n")  if(abs(afterpar-beforepar)<10^-6){  conv<-TRUE  cat("the final estimator of theta is:",afterpar,"\n")  break} else{loop<-loop+1;beforepar<-afterpar;before.y3t<-(beforepar/4)/(beforepar/4+1/2)}  } |

2.

|  |
| --- |
| rm(list=ls())  graphics.off()  ######  n<-5000  f<-function(x,u,sigma){  (1/sqrt(2\*pi\*sigma))\*exp(-(x-u)^2/(2\*sigma))  }  compute.z1<-function(x,p1,p2,p3,u1,sigma1,u2,sigma2,u3,sigma3){  y<-p1\*f(x,u1,sigma1)/(p1\*f(x,u1,sigma1)+p2\*f(x,u2,sigma2)+p3\*f(x,u3,sigma3))  return(y)  }  compute.z2<-function(x,p1,p2,p3,u1,sigma1,u2,sigma2,u3,sigma3){  y<-p2\*f(x,u2,sigma2)/(p1\*f(x,u1,sigma1)+p2\*f(x,u2,sigma2)+p3\*f(x,u3,sigma3))  return(y)  }  compute.z3<-function(x,p1,p2,p3,u1,sigma1,u2,sigma2,u3,sigma3){  y<-p3\*f(x,u3,sigma3)/(p1\*f(x,u1,sigma1)+p2\*f(x,u2,sigma2)+p3\*f(x,u3,sigma3))  return(y)  }  generate.normal<-function(n){  Z<-numeric(n)  conv<-FALSE  loop<-0  while(!conv){  u1<-runif(1,0,1)  u2<-runif(1,0,1)  u3<-runif(1,0,1)  y<- -log(u1)  if(u2<=exp(-(y-1)^2/2)){  if(u3<1/2){loop<-loop+1;Z[loop]<- -y} else{loop<-loop+1;Z[loop]<-y}  }  if(loop==n){conv<-TRUE}  }  return(Z)  }  x1<-generate.normal(n\*0.5)\*1+2  x2<-generate.normal(n\*0.3)\*3+3  x3<-generate.normal(n\*0.2)  x<-c(x2,x3,x1)  before.p1<-0.22  before.p2<-0.28  before.p3<-0.5  before.u1<-10  before.u2<-5  before.u3<-3  before.sigma1<-10  before.sigma2<-5  before.sigma3<-1/2  before.par<-c(before.p1,before.p2,before.p3,before.u1,before.u2,before.u3,before.sigma1,before.sigma2,before.sigma3)  loop<-1  conv<-FALSE  while(!conv){  zz1<-numeric(n);zz2<-numeric(n);zz3<-numeric(n);zz1.sigma<-numeric(n);zz2.sigma<-numeric(n);zz3.sigma<-numeric(n);zz1.u<-numeric(n);zz2.u<-numeric(n);zz3.u<-numeric(n)  for(i in 1:n){  z1<-compute.z1(x[i],before.p1,before.p2,before.p3,before.u1,before.sigma1,before.u2,before.sigma2,before.u3,before.sigma3)  zz1[i]<-z1  zz1.u[i]<-z1\*x[i]  z2<-compute.z2(x[i],before.p1,before.p2,before.p3,before.u1,before.sigma1,before.u2,before.sigma2,before.u3,before.sigma3)  zz2[i]<-z2  zz2.u[i]<-z2\*x[i]  z3<-compute.z3(x[i],before.p1,before.p2,before.p3,before.u1,before.sigma1,before.u2,before.sigma2,before.u3,before.sigma3)  zz3[i]<-z3  zz3.u[i]<-z3\*x[i]  }  after.p1<-sum(zz1)/n  after.p2<-sum(zz2)/n  after.p3<-sum(zz3)/n  after.u1<-sum(zz1.u)/sum(zz1)  after.u2<-sum(zz2.u)/sum(zz2)  after.u3<-sum(zz3.u)/sum(zz3)  z11<-numeric(n);z22<-numeric(n);z33<-numeric(n)  for(i in 1:n){  z1<-compute.z1(x[i],before.p1,before.p2,before.p3,before.u1,before.sigma1,before.u2,before.sigma2,before.u3,before.sigma3)  zz1.sigma[i]<-z1\*(x[i]-after.u1)^2  z11[i]<-compute.z1(x[i],before.p1,before.p2,before.p3,before.u1,before.sigma1,before.u2,before.sigma2,before.u3,before.sigma3)    z2<-compute.z2(x[i],before.p1,before.p2,before.p3,before.u1,before.sigma1,before.u2,before.sigma2,before.u3,before.sigma3)  zz2.sigma[i]<-z2\*(x[i]-after.u2)^2  z22[i]<-compute.z2(x[i],before.p1,before.p2,before.p3,before.u1,before.sigma1,before.u2,before.sigma2,before.u3,before.sigma3)    z3<-compute.z3(x[i],before.p1,before.p2,before.p3,before.u1,before.sigma1,before.u2,before.sigma2,before.u3,before.sigma3)  zz3.sigma[i]<-z3\*(x[i]-after.u3)^2  z33[i]<-compute.z3(x[i],before.p1,before.p2,before.p3,before.u1,before.sigma1,before.u2,before.sigma2,before.u3,before.sigma3)  }  after.sigma1<-sum(zz1.sigma)/sum(z11)  after.sigma2<-sum(zz2.sigma)/sum(z22)  after.sigma3<-sum(zz3.sigma)/sum(z33)    after.par<-c(after.p1,after.p2,after.p3,after.u1,after.u2,after.u3,after.sigma1,after.sigma2,after.sigma3)    cat("now is:",loop,"th loop and nth par. is:",before.par,"n+1th is",after.par,"\n")  cat("the error is:",max(abs(after.par-before.par)),"\n\n")  if(max(abs(after.par-before.par))<10^-5){conv<-TRUE} else{  loop<-loop+1  before.p1<-after.p1  before.p2<-after.p2  before.p3<-after.p3  before.u1<-after.u1  before.u2<-after.u2  before.u3<-after.u3  before.sigma1<-after.sigma1  before.sigma2<-after.sigma2  before.sigma3<-after.sigma3  before.par<-after.par  }    } |

3.

|  |
| --- |
| rm(list=ls())  graphics.off()  #####  compare.MH<-function(x0,error,n){  xn<-x0  collect<-numeric(n)  collect[1]<-xn  conv<-FALSE  loop<-1  while(!conv){  loop<-loop+1  y<-runif(1,xn-error,xn+error)  min.xy<-min(exp(-1/2\*y^2+1/2\*xn^2),1)  u<-runif(1,0,1)  if(u<=min.xy){xn<-y} else{xn<-xn}  collect[loop]<-xn  if(loop==n){conv<-TRUE}  }  plot(collect,type="l",col="red",main = paste("X0 =",x0,"epsilon =",error))  return(collect)  }  a<-compare.MH(0,1,10000) |