Take home quesiton

Muzhou liu

October 26, 2018

a)

```
D_n <- function(theta, q , data ){
  n<- length(data)
  return((sum(abs(theta-data)-data)/(2*n) + (1-2*q)*theta/2)*0.05)
}
```

b)

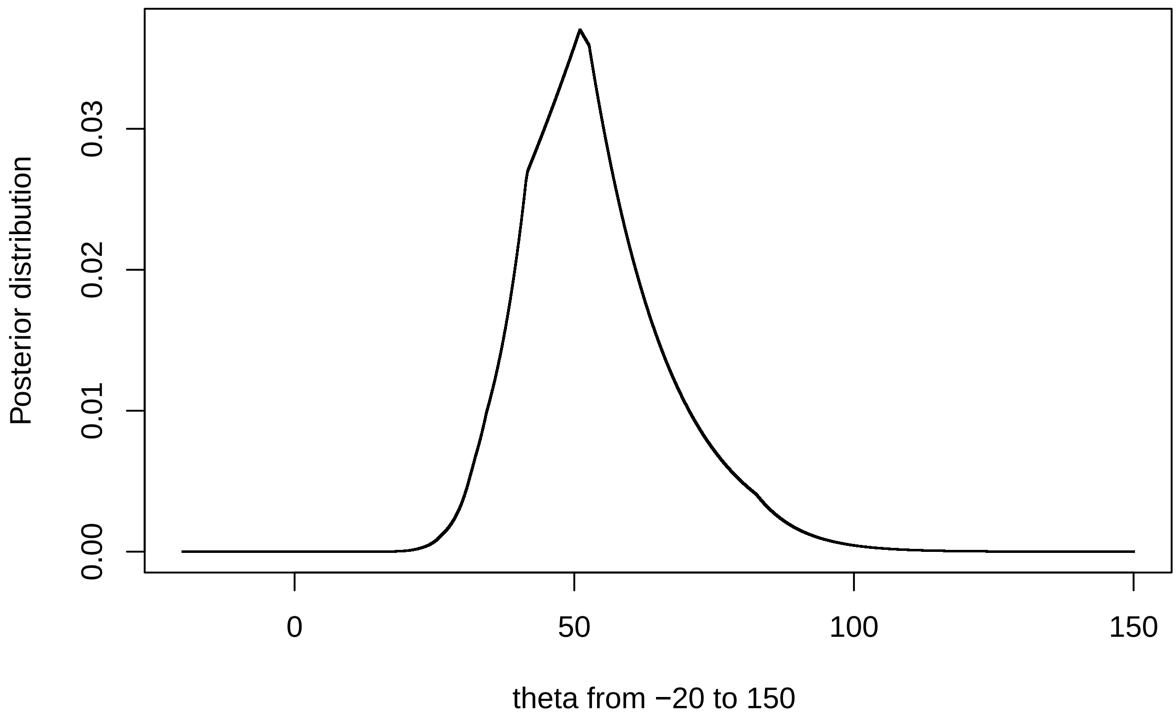
```
norwegian.fire <- fread('https://www.math.wustl.edu/~nasyring/475/norwegianfire.txt')
X.old <- norwegian.fire$V1[norwegian.fire$V2 == 89]/500
 X <- norwegian.fire$V1[norwegian.fire$V2 == 90]/500
```

c)

```
p.shape <- 2
p.scale <- as.numeric(quantile(X.old, 0.995)/2)


numerator <- function(theta,q, data){return(exp(-length(data)*D_n(theta,0.995,data))*
                                              dgamma(theta,shape = p.shape,scale = p.scale))}


Simpson_rule_exam <- function(fun, up, lo, n, q, data){
   if(lo>up){
     c <- up
     up <- lo
     lo <- c
   }
   xi_s <- lo+ c(1:n)/n *(up-lo)
   h <- (up-lo)/n
   sum <-0
   for (i in 1:(n-1)) {
     sum <- sum+ h/6 * (fun(xi_s[i],q,data)+4* fun((xi_s[i]+xi_s[i+1])/2,q,data) +
                          fun(xi_s[i+1],q,data))
   }

   return(sum)
}

denominator <- Simpson_rule_exam(numerator,-1000,1000,10000,q,X)


posterior_exam <- function(theta, q,data) {
  numerator(theta,q,data)/ denominator
}
```

d)

```
theta_is <- seq(-20,150,by = 0.001) %>% data.frame()

apply(theta_is,1, function(x){posterior_exam(x,0.995,X)}) -> temp_result

plot(seq(-20,150,by = 0.001),temp_result,pch='.', xlab = 'theta from -20 to 150 ', ylab = 'Posterior di
```



e)

```
# using the gradient descend to find the minimum of negative posterior distribution

gradent_neg_posterior <- function(theta,q,data){

  dn <- D_n(theta,q,data)
  n <- length(data)

  return(-(-n*exp(-n*dn)*(sum(sign(theta-data))/(2*n) + (1-2*q)/2)*0.05*
             dgamma(theta, scale = p.scale, shape = p.shape)+
             ((p.shape-1)/theta^2 - 1/(theta*p.scale))*
             theta^(p.shape)*exp(-theta/p.scale) *exp(-n*dn) *
             1/(factorial(p.shape-1)*p.scale^p.shape))/denominator)
}
```

```
graden_descend <- function(init_theta,grad,tol, max_it, learning_rate ,q, data){
  old_theta <- init_theta
  n_it <- 0
  rea_tol <- tol*2
  while (rea_tol >tol & n_it < max_it) {
    new_theta <- old_theta - grad(old_theta,q,data)*learning_rate
    rea_tol <- abs(grad(new_theta,q,data))
    n_it <- n_it+1
    old_theta <- new_theta
    if(grad(new_theta,q,data)>grad(old_theta,q,data)){learning_rate = learning_rate/2}

  }
  return(list(solution = new_theta, n_it = n_it, last_tol = rea_tol))
}


graden_descend(40, gradent_neg_posterior, 10^(-4), 10000,1, 0.995, X)
```

```
## $solution
## [1] 51.0183488470972
##
## $n_it
## [1] 10000
##
## $last_tol
## [1] 0.000676835688413607
```