Homework2_Optimization

Xueying_Li

2/7/2018

Question1

(a)

First,

$$p(x;\theta) = \frac{1}{\pi[1+(x-\theta)^2]}$$

Since $x_1, ..., x_n$ is an i.i.d.

$$\begin{aligned}
l(\theta) &= \ln(\prod_{i=1}^{n} p(x_i;\theta)) \\
&= \ln(\prod_{i=1}^{n} \frac{1}{\pi[1+(x_i-\theta)^2]}) \\
&= \sum_{i=1}^{n} [\ln\frac{1}{\pi} + \ln\frac{1}{1+(x-\theta)^2}] \\
&= -n\ln\pi - \sum_{i=1}^{n} \ln[1+(\theta-x_i)^2]
\end{aligned}$$

Then we calculated the first derivative

$$l'(\theta) = -\sum_{i=1}^{n} \frac{2(\theta-x_i)}{1+(\theta-x_i)^2} = -2\sum_{i=1}^{n} \frac{\theta-x_i}{1+(\theta-x_i)}$$

The second derivative is as followings

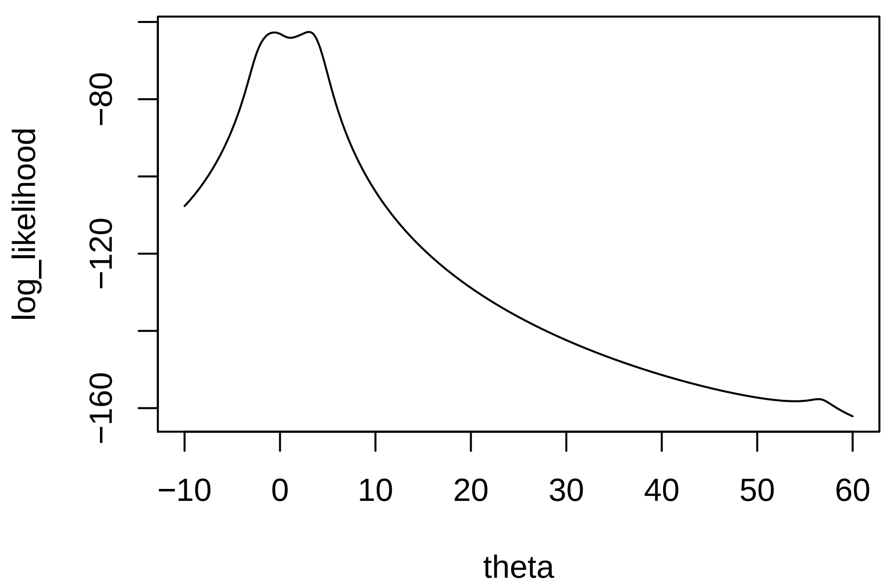
$$l''(\theta) = -2\sum_{i=1}^{n} \frac{1+(\theta-x_i)^2-2(\theta-x_i)(\theta-x_i)}{[1+(\theta-x_i)^2]^2} = -2\sum_{i=1}^{n} \frac{1-(\theta-x_i)^2}{[1+(\theta-x_i)^2]^2}$$

The fisher score is as followings

$$
\begin{aligned}
I(\theta) &= n\int \frac{\{p'(x)\}^2}{p(x)}dx \\
&= n\int \frac{4(x-\theta)^2\pi[1+(x-\theta)^2]}{\pi[1+(x-\theta)^2]^4}dx \\
&= \frac{4n}{\pi}\int_{-\infty}^{\infty}\frac{x^2}{(1+x^2)^3}dx \\
&= \frac{4n}{\pi}\int_{-\infty}^{\infty}[(\frac{1}{(1+x^2)^2}-\frac{1}{(1+x^2)^3})]dx \\
&= \frac{4n}{\pi}(\int_{-\infty}^{\infty}\frac{1}{(1+x^2)^2}dx-\int_{-\infty}^{\infty}\frac{1}{(1+x^2)^3}dx) \\
&= \frac{4n}{\pi}[\frac{1}{4}(\frac{x}{2(x^2+1)}|_{-\infty}^{\infty}+\frac{1}{2}\int_{-\infty}^{\infty}\frac{1}{1+x^2}dx)-\frac{x}{4(x^2+1)^2}|_{-\infty}^{\infty}] \\
&= \frac{4n}{\pi}(\frac{x(x^2-1)}{8(x^2+1)^2}|_{-\infty}^{\infty}+\frac{1}{8}\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}}\frac{\sec^2 t}{1+\tan^2 t}dt) \\
&= \frac{4n}{\pi}(0+\frac{\pi}{8}) \\
&= \frac{n}{2}
\end{aligned}
$$

(b)

The graph of the log-likelihood is as followings



Then we wrote the function for Newton-Raphson method. Here are the codes and the results:

```
deriv_llf <- function(theta){
  deriv <- 0
```

```r
  for(i in 1:length(observed_sample))
    deriv <- deriv +
      (-2) * ((theta - observed_sample[i])/(1 + (theta - observed_sample[i])^2))
  return(deriv)
}

hessian_llf <- function(theta){
  hess <- 0
  for(i in 1:length(observed_sample)){
    temp <- (1 - (theta - observed_sample[i])^2)/(1 + (theta - observed_sample[i])^2)^2
    hess <- hess + (-2) * sum(temp)
  }
  return(hess)
}

nr_method <- function(a){
  theta <- array()
  theta[1] <- a
  #initialize the h_t equal to 1 in order to begin the loop
  h_t <- 1
  i <- 1
  #we set the max loop number be 100 and the upper bound of error be 0.0001
  while(abs(h_t) > 0.0001 & i<100){
    h_t <- (-1)*deriv_llf(theta[i])/hessian_llf(theta[i])
    theta[i+1] <- theta[i] + h_t
    i <- i+1
  }
  return(theta[i])
}
# a <- nr_method(7)
# a
nr_method_value <- sapply(start_value, nr_method)
nr_method_value
```

```
##  [1] -3.660385e+30 -5.914735e-01 -5.914735e-01  1.092730e+00  3.021345e+00
##  [6] -5.914735e-01  1.527426e+30  8.084521e+29  1.135250e+31  3.021345e+00
```

(c)

Similarly, we wrote the function for fixed point method and fisher score method

```r
##question 1_c----------------------------------
fixpoint_method <- function(a,alpha){
  theta <- array()
  theta[1] <- a
  h_t <- 1
  i <- 1
  while(abs(h_t) > 0.0001 & i<1000){
    h_t <- alpha*deriv_llf(theta[i])
    theta[i+1] <- theta[i] + h_t
    i <- i+1
  }
  return(theta[i])
}
```

```r
a1 <- fixpoint_method_value <- sapply(start_value,fixpoint_method, alpha = alpha[1])
a2 <- fixpoint_method_value <- sapply(start_value,fixpoint_method, alpha = alpha[2])
a3 <- fixpoint_method_value <- sapply(start_value,fixpoint_method, alpha = alpha[3])
a1
```

```
##  [1] -0.5914254  0.1035079 -1.1063091  0.1035079 -1.1063091 -1.1713919
##  [7] -1.1713919  0.2417269  0.2417269 -1.1063091
```

```r
a2
```

```
##  [1] -0.5914836 -0.5914824 -0.5914659  3.2398379 -0.5914671 -0.5914885
##  [7]  2.5915177 -0.5914885  2.5915177  2.5915177
```

```r
a3
```

```
##  [1] -0.5915732 -0.5915348 -0.5913732  3.0213435  3.0213445  3.0213440
##  [7]  3.0213452  3.0213442  3.0213435  3.0213451
```

(d)

```r
##question 1_d-------------------------------
fisherscore_method <- function(a){
  theta <- array()
  theta[1] <- a
  #initialize the h_t equal to 1 in order to begin the loop
  h_t <- 1
  i <- 1
  #we set the max loop number be 100 and the upper bound of error be 0.0001
  while(abs(h_t) > 0.01 & i<1000){
    h_t <- deriv_llf(theta[i])/(length(observed_sample) / 2)
    theta[i+1] <- theta[i] + h_t
    i <- i+1
  }
  return(theta[i])
}

fisherscore_method_value <- sapply(start_value, fisherscore_method)
#refine the value by newton raphson method
refine_value <- sapply(fisherscore_method_value,nr_method)
refine_value
```

```
##  [1] -0.5914735 -0.5914735 -0.5914735  3.0213454  3.0213454  3.0213455
##  [7]  3.0213454  3.0213454  3.0213454  3.0213454
```

(e)

We can easily find that Newton's method sometimes counld not get to the converge point when the start points are not good enough. For the Fix-point method, the benefit is its fast speed for iteration while it is not precise as the Newton's method. Generally speaking, we could use the Fix-point method to get the relatively good start point and then refine it by the Newton's method.

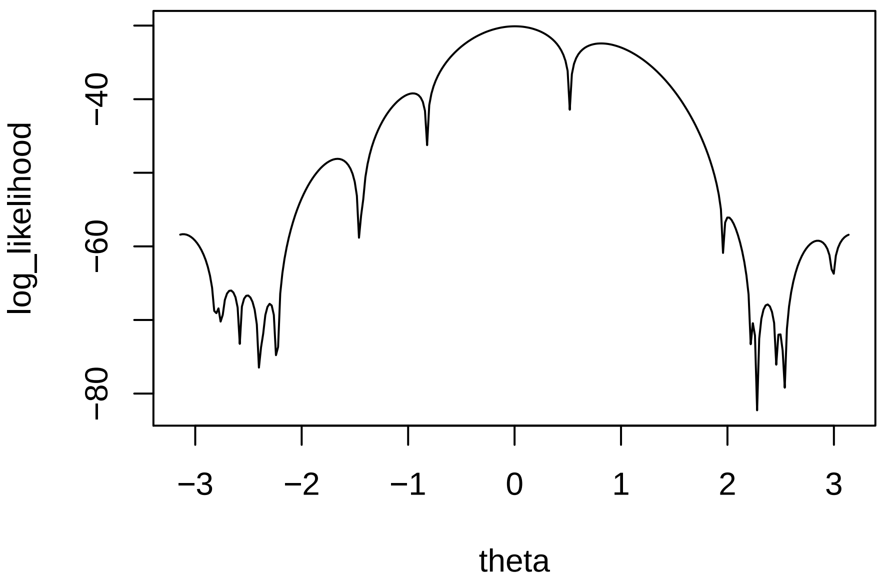
Question2

(a)

The log-likelihood function of θ is

$$-n \ln 2\pi + \sum_{i=1}^{n} \ln[1 - \cos(x_i - \theta)]$$

The graph of it is as followings



(b)

We can easily calculate the $\int_0^{2\pi} xP(x)dx$ and get that

$$E[x|\theta] = \pi + \sin(\theta) \tag{1}$$

Then we used the function uniroot to get the results

```
## [1] 0.09539408
```

```
## [1] 3.046199
```

(c)

The methodology is the same as the one in question1.

```
##question 2-c---------------------------------
start_value <- c(r1,r2)

deriv_llf <- function(a){
  deriv <- sum(sin(observed_sample-a)/(1-cos(observed_sample-a)))
  return(deriv)
```

```r
}

hessian_llf <- function(a){
  hess <-(sum(1/(1-cos(observed_sample-a))))
  return(hess)
}

nr_method <- function(a){
  theta <- array()
  theta[1] <- a
  #initialize the h_t equal to 1 in order to begin the loop
  h_t <- 1
  i <- 1
  #we set the max loop number be 100 and the upper bound of error be 0.0001
  while(abs(h_t) > 0.0001 & i<1000){
    h_t <- (-1)*deriv_llf(theta[i])/hessian_llf(theta[i])
    theta[i+1] <- theta[i] + h_t
    i <- i+1
  }
  return(theta[i])
}

nr_method_value <- sapply(start_value, nr_method)
nr_method_value
```

```
## [1] 0.003118157 3.170714800
```

(d)

```r
##question 2-d-------------------------------
nr_method(2.7)
```

```
## [1] 2.848415
```

```r
nr_method(-2.7)
```

```
## [1] -2.668857
```

(e)

We divide the interval into 200 pieces and get the converge value for each point. As for the last table, we should notice that the first interval is from $-\pi$ to the first value in the table, which means the interval from the first value to the second value will converge the second data in nt_value. nr_value represents the converge value.

```r
##question 2-e-------------------------------

start_value <- seq(-pi,pi,length.out = 200)
nr_method_value <- sapply(start_value, nr_method)
nr_value <- round(nr_method_value,6)
freq <- as.data.frame(table(nr_value))
interval_index <- array()
interval_index[1] <- freq$Freq[1]
for(i in 2:length(freq$Freq)){
  interval_index[i] <- freq$Freq[i]+interval_index[i-1]
```

```
}
#freq[1]
internal <- c()
for(i in 1:length(interval_index)){
  internal[i] <- start_value[interval_index[i]]
}
#internal
table_converge <- cbind(freq[1],internal)
table_converge
```

```
##     nr_value   internal
## 1  -3.112471 -2.8258547
## 2  -2.786557 -2.7627071
## 3  -2.668857 -2.6048381
## 4  -2.509356 -2.4153954
## 5  -2.388267 -2.3838216
## 6  -2.297926 -2.2575264
## 7  -2.232192 -2.2259526
## 8  -1.662712 -1.4681815
## 9  -1.447503 -1.4366077
## 10 -0.954406 -0.8367056
## 11  0.003118  0.4893938
## 12  0.812637  1.9417884
## 13  2.007223  2.1943788
## 14  2.237013  2.2575264
## 15  2.374712  2.4469692
## 16   2.48845  2.5101167
## 17  2.848415  2.9837237
## 18  3.170715  3.1415927
```

Question3

(a)

The difference from the first two questions is that $\theta = [K, r]$. So every time we calculate the function with two parameters. Here are our codes:

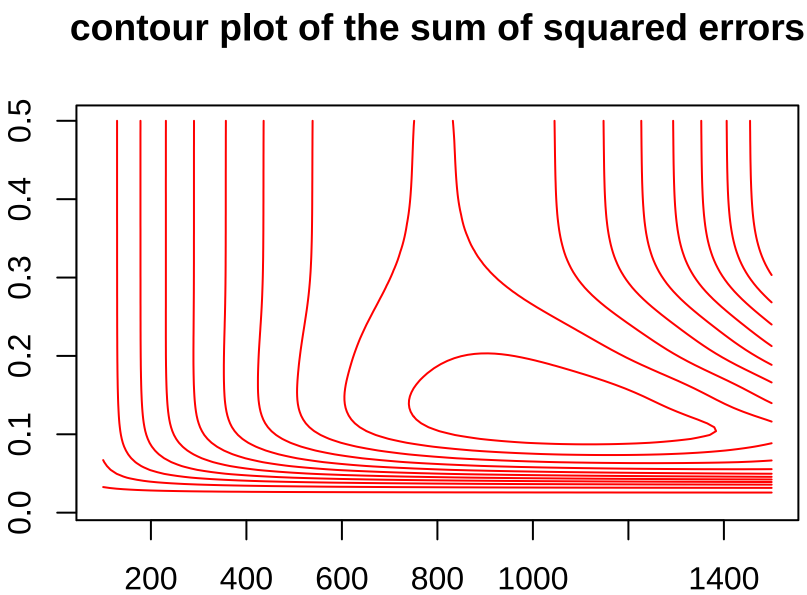
```
#rm(list = ls())
beetles <- data.frame( days = c(0, 8, 28, 41, 63, 69, 97, 117, 135, 154),
                       beetles = c(2, 47, 192, 256, 768, 896, 1120, 896, 1184, 1024))
N0 <- beetles$beetles[1]
n <- length(beetles$days)
A_calculate <- function(k,r){
  A <- matrix(0,nrow = n, ncol = 2)
  for(i in 1:n){
    temp1 <- (N0^2-N0^2*exp(-1*r*beetles$days[i]))
    A[i,1] <- temp1/((N0+(k-N0)*exp(-1*r*beetles$days[i]))^2)
    temp <- beetles$days[i]*(k-N0)*exp(-1*r*beetles$days[i])*k*N0
    A[i,2] <- temp/((N0+(k-N0)*exp(-1*r*beetles$days[i]))^2)
  }
  return(A)
}

Z_calculate <- function(k,r){
```

```r
  Z <- matrix(0,nrow = n, ncol = 1)
  for(i in 1:n){
    Z[i,1] <- beetles$beetles[i] - k*N0/(N0+(k-N0)*exp(-1*r*beetles$days[i]))
  }
  return(Z)
}

h_t_calculate <- function(k,r){
 A <- A_calculate(k,r)
 Z <- Z_calculate(k,r)
 h_t <- solve(t(A) %*% A) %*% t(A) %*% Z
 return(h_t)
}

##question 3-a----------------------------
gauss_newton_method <- function(k0,r0){
  h_t <- c(1000,0.1)
  error <- sum(abs(h_t))
  error_modified_a <- 2
  error_modified_b <- 1
  i <- 1
  while(abs(error_modified_b - error_modified_a) > 0.00000001 & i<1000){
    error_modified_a <- error_modified_b
    h_t <- h_t_calculate(k0,r0)
    k0 <- k0+h_t[1]
    r0 <- r0+h_t[2]
    error <- sum(abs(h_t))
    error_modified_b <- sum(abs(h_t))
    i <- i+1
    #print(h_t)
  }
  #i is the iteration times
  theta <- c(k0,r0,i)
  return(theta)
}

a <- gauss_newton_method(1200,0.5)
a
```

```
## [1] 1049.4072443    0.1182684   18.0000000
```

(b)

```
## Warning: package 'rgl' was built under R version 3.4.3
```



(c)

If $log(N_t)$ is log-normal, we know that $P(log(N_t)) = \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(log(N_t)-log(f_i))^2}{2\sigma^2})$. Then We derive the log-likelihood function by take the sum of $log(P(N_t))$. We use Fixed_point Method to optimize $\theta = [K, r, \sigma]$. In this way, we need to calcualte the derivative for each parameter. Also, we set three different value for α responding to three parameters.

```
##question 3-c-----------------------------
l_k_derivative <- function(k,r,sigma){
  l_k_derivative_value <- 0
  for(i in 1:n){
    temp1 <- ((N0+(k-N0)*exp(-1*r*beetles$days[i]))^2)
    temp <- (N0^2-N0^2*exp(-1*r*beetles$days[i]))/temp1
    temp2 <- (log(beetles$beetles[i])-log(k*N0/(N0+(k-N0)*exp(-1*r*beetles$days[i]))))
    temp3 <- (1/(sigma*sigma))*temp2/(k*N0/(N0+(k-N0)*exp(-1*r*beetles$days[i])))*temp
    l_k_derivative_value <- l_k_derivative_value + temp3
  }
  return(l_k_derivative_value)
}
#l_r_derivative(1000,0.1,1)
l_r_derivative <- function(k,r,sigma){
  l_r_derivative_value <- 0
  for(i in 1:n){
    temp1 <- ((N0+(k-N0)*exp(-1*r*beetles$days[i]))^2)
    temp <- beetles$days[i]*(k-N0)*exp(-1*r*beetles$days[i])*k*N0/temp1
    temp2 <- (log(beetles$beetles[i])-log(k*N0/(N0+(k-N0)*exp(-1*r*beetles$days[i]))))
    temp3 <- (1/(sigma*sigma))*temp2/(k*N0/(N0+(k-N0)*exp(-1*r*beetles$days[i])))*temp
    l_r_derivative_value <- l_r_derivative_value + temp3
  }
```

```r
  return(l_r_derivative_value)
}

l_sigma_derivative <- function(k,r,sigma){
  l_sigma_derivative_value <- 0
  for(i in 1:n){
    temp <- k*N0/(N0+(k-N0)*exp(-1*r*beetles$days[i]))
    temp2 <- ((log(beetles$beetles[i])-log(temp))^2)/(sigma^3)
    l_sigma_derivative_value <- l_sigma_derivative_value + temp2
  }
  l_sigma_derivative_value <- l_sigma_derivative_value - n/(2*sigma)
  return(l_sigma_derivative_value)
}
#l_sigma_derivative(1000,0.1,1)

l_derivative <- function(k,r,sigma){
  l_derivative_value <- c()
  l_derivative_value[1] <- l_k_derivative(k,r,sigma)
  l_derivative_value[2] <- l_r_derivative(k,r,sigma)
  l_derivative_value[3] <- l_sigma_derivative(k,r,sigma)
  return(l_derivative_value)
}
#a <- l_derivative(1000,0.1,1)

fixpoint_method <- function(k0,r0,sigma0){
  h_t <- c(1000,1,1)
  error <- sum(abs(h_t))
  i <- 1
  while(error>0.001 & i<3000){
    h_t <- l_derivative(k0,r0,sigma0)
    #result <- c(k0,r0,sigma0,i)
    #print(result)
    k0 <- k0 + h_t[1] * 1000#rnorm(1, 5000000, 1000000)
    r0 <- r0+h_t[2] * 0.0003#rnorm(1, 0.0005, 1)
    sigma0 <- sigma0+h_t[3]* 0.0003#rnorm(1, 0.05, 0.01)
    error <- sum(abs(h_t))
    i <- i+1

    #print(error)
  }
  result <- c(k0,r0,sigma0,i)
  return(result)
}
a <- fixpoint_method(800,0.15,0.1)
a
```

```
## [1]  672.0907892    0.4004609    0.8286135 2773.0000000
```

As for the variance, we document the error every steps and calcualte its variance.

```
## [1] 2.37656
```