Math5361 Homework II

Huiqiong Liao

2/8/2018

Question 1

(a)

Given the density function of $Cauchy(x;\theta)$:

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

Where $x_1, x_2...x_n$ are i.i.d. sample and $l(\theta)$ equals to:

$$l(\theta) = \ln(\prod_1^n p(x_i;\theta)) = \sum_1^n \ln(\frac{1}{\pi[1+(x_i-\theta)^2]})$$
$$= \sum_1^n \ln(\frac{1}{\pi}) + \sum_1^n \ln(\frac{1}{1+(x_i-\theta)^2}) = -n\ln\pi - \sum_1^n \ln[1+(\theta-x_i)^2]$$

Next, to prove $l^{'}(\theta)$,

$$l^{'}(\theta) = 0 - [\sum_1^n \ln(1+(\theta-x_i)^2)]^{'}_{\theta} = -2\sum_1^n \frac{\theta-x_i}{1+(\theta-x_i)^2}$$

Then, to prove the second derivative $l^{''}(\theta)$, do the derivative of $l^{'}(\theta)$,

$$l^{''}(\theta) = -2\sum_1^n (\frac{\theta-x_i}{1+(\theta-x_i)^2})^{'}_{\theta} = -2\sum_1^n \frac{1-{x_i}^2-\theta^2+2x_i\theta}{[1+(\theta-x_i)^2]^2} = -2\sum_1^n \frac{1-(\theta-x_i)^2}{[1+(\theta-x_i)^2]^2}$$

The final step is to calculate the fisher score $I(\theta)$, do the integration:

$$I(\theta) = n\int_{-\infty}^{\infty} \frac{[p^{'}(x)]^2}{p(x)}dx = \frac{4n}{\pi}\int_{-\infty}^{\infty} \frac{x^2}{(1+x^2)^3}dx = \frac{4n}{\pi}[\int_{-\infty}^{\infty} \frac{1}{(1+x^2)^2}dx - \int_{-\infty}^{\infty} \frac{1}{(1+x^2)^3}dx]$$

Then,apply the $In = \int \frac{1}{(x^2+a^2)^n}dx$ method, do the integration by parts:

$$\int \frac{1}{(1+x^2)^2}dx = \frac{x}{2(x^2+1)} + \frac{1}{2}\int \frac{1}{x^2+1}dx = \frac{x}{2(x^2+1)} + \frac{1}{2}\arctan(x)$$

And the second part equals to:

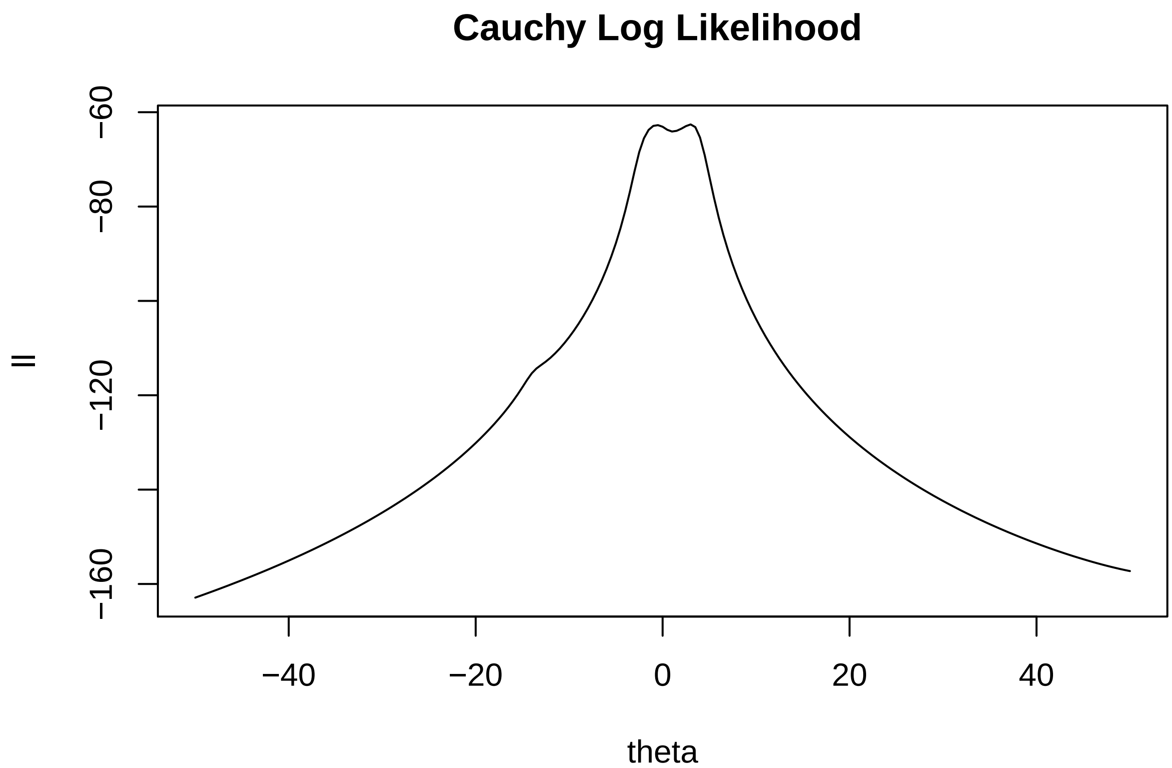
$$\int \frac{1}{(x^2+1)^3}dx = \frac{3}{8}(\frac{x}{x^2+1}) + \frac{3}{8}\arctan(x) + \frac{x}{4(x^2+1)^2}$$

Add them together to get the final answer:

$$I(\theta) = \frac{4n}{\pi}(\frac{\arctan(x)}{8}|_{-\infty}^{+\infty} + \frac{(x^3-x)}{8(x^2+1)^2}|_{-\infty}^{+\infty}) = \frac{4n}{\pi}(\frac{\pi}{8}) + 0 = \frac{n}{2}$$

(b) Graph the likelihood function

```
x <- c(1.77, -0.23, 2.76, 3.80, 3.47, 56.75, -1.34, 4.24, -2.44,
       3.29, 3.71, -2.40, 4.53, -0.07, -1.05, -13.87, -2.53, -1.75)
th <- seq(-50,50,0.5)
n <- length(x)
ll <- c()
for (i in 1:length(th)) {
  ll[i] <- -n*log(pi)-sum(log(1+(th[i]-x)^2))
}
plot(th,ll,type = "l", ylab = "ll", xlab = "theta",
main = "Cauchy Log Likelihood ")
```



Created a NewtonR function to find MLE for each starting point.

```
#find MLE for theta using the Newton-Raphson method
cauchy.ll.1st.deriv <- function(theta){
  z <- -2*(sum((theta-x)/(1+(theta-x)^2)))
  return(z)
}
cauchy.ll.2nd.deriv <- function(theta){
  y <- -2*(sum((1-(theta-x)^2)/((1+(theta-x)^2)^2)))
  return(y)
}
staring.p <- c(-11,-1, 0, 1.5, 4, 4.7, 7, 8, 38)
sample_mean <- mean(staring.p)

NewtonR <- function(theta0){
  theta <- array()
  theta[1] <- theta0
  i <- 1
  difference <- 10
  while (abs(difference)>10^(-10) & i<200){
    theta[i+1] <- theta[i] -
      cauchy.ll.1st.deriv(theta[i]) /cauchy.ll.2nd.deriv(theta[i])
```

```r
    difference <- abs(theta[i+1] - theta[i])
    i <- i + 1
  }
  return(theta[i])
}
NewtonR(-11)
```

```
## [1] -4.640089e+60
```

```r
NewtonR(-1)
```

```
## [1] -0.5914735
```

```r
NewtonR(0)
```

```
## [1] -0.5914735
```

```r
NewtonR(1.5)
```

```
## [1] 1.09273
```

```r
NewtonR(4)
```

```
## [1] 3.021345
```

```r
NewtonR(7)
```

```
## [1] 1.936242e+60
```

```r
NewtonR(8)
```

```
## [1] 1.024835e+60
```

```r
NewtonR(38)
```

```
## [1] 1.439101e+61
```

```r
NewtonR(sample_mean)
```

```
## [1] 1.358409e+60
```

```r
# When starting at -11,7,8,and 38, the function cannot find the optimial point
# Sample mean(5.68889) also is not a good starting point
```

(c)Fixed-point iteration

Given $G(x) = \alpha l^{'}(\theta) + \theta$ with $\alpha_1 = 1$, $\alpha_2 = 0.64$, and $\alpha_3 = 0.25$ Using $\theta_{t+1} = \theta_t + \alpha l^{'}(\theta_t)$ to do iteration.

The first case: when $\alpha = 1$

```r
#alpha = 1, 0.64 or 0.25
#alpha <- -1 / cauchy.ll.2nd.deriv(theta)
alpha1 <- 1
fixP1 <- function(theta0){
  theta <- array()
  theta[1] <- theta0
  i <- 1
  difference <- 10
  while (abs(difference)>10^(-10) & i<200) {
    theta[i+1] <- theta[i] + alpha1 * cauchy.ll.1st.deriv(theta[i])
    difference <- abs(theta[i+1] - theta[i])
    i <- i + 1
  }
  return(theta[i])
}

fixP1(-11)
```

```
## [1] -0.591472
```

```r
fixP1(-1)
```

```
## [1] 0.1035079
```

```r
fixP1(0)
```

```
## [1] -1.106309
```

```r
fixP1(1.5)
```

```
## [1] 0.1035079
```

```r
fixP1(4)
```

```
## [1] -1.106309
```

```r
fixP1(4.7)
```

```
## [1] -1.171392
```

```r
fixP1(7)
```

```
## [1] -1.171392
```

```r
fixP1(8)
```

```
## [1] 0.2417269
```

```r
fixP1(38)
```

```
## [1] 0.2417269
```

The second case, when $\alpha = 0.64$

```r
alpha2 <- 0.64
fixP2 <- function(theta0){
  theta <- array()
  theta[1] <- theta0
  i <- 1
  difference <- 10
  while (abs(difference)>10^(-10) & i<200) {
    theta[i+1] <- theta[i] + alpha2 * cauchy.ll.1st.deriv(theta[i])
    difference <- abs(theta[i+1] - theta[i])
    i <- i + 1
  }
  return(theta[i])
}

fixP2(-11)
```

```
## [1] -0.5914735
```

```r
fixP2(-1)
```

```
## [1] -0.5914735
```

```r
fixP2(0)
```

```
## [1] -0.5914735
```

```r
fixP2(1.5)
```

```
## [1] 3.239838
```

```r
fixP2(4)
```

```
## [1] -0.5914735
```

```r
fixP2(4.7)
```

```
## [1] -0.5914735
```

```r
fixP2(7)
```

```
## [1] 2.591518
```

```r
fixP2(8)
```

```
## [1] -0.5914735
```

```r
fixP2(38)
```

```
## [1] 2.591518
```

The third case, when $\alpha = 0.25$

```r
alpha3 <- 0.25
fixP3 <- function(theta0){
  theta <- array()
  theta[1] <- theta0
  i <- 1
  difference <- 10
  while (abs(difference)>10^(-10) & i<200) {
    theta[i+1] <- theta[i] + alpha3 * cauchy.ll.1st.deriv(theta[i])
    difference <- abs(theta[i+1] - theta[i])
    i <- i + 1
  }
  return(theta[i])
}

fixP3(-11)
```

```
## [1] -0.5914735
```

```r
fixP3(-1)
```

```
## [1] -0.5914735
```

```r
fixP3(0)
```

```
## [1] -0.5914735
```

```r
fixP3(1.5)
```

```
## [1] 3.021345
```

```r
fixP3(4)
```

```
## [1] 3.021345
```

```r
fixP3(4.7)
```

```
## [1] 3.021345
```

```r
fixP3(7)
```

```
## [1] 3.021345
```

```r
fixP3(8)
```

```
## [1] 3.021345
```

```r
fixP3(38)
```

```
## [1] 3.021345
```

(d)Fisher Score

```
fisher.score <- n / 2
FisherFun <- function(theta0){
  theta <- array()
  theta[1] <- theta0
  i <- 1
  difference <- 10
  while (abs(difference)>10^(-10) & i<200){
    theta[i+1] <- theta[i] + cauchy.ll.1st.deriv(theta[i]) / (fisher.score)
    difference <- abs(theta[i+1] - theta[i])
    i <- i + 1
  }
  return(theta[i])
}
FisherFun(-11)
```

```
## [1] -0.5914735
```

```
FisherFun(-1)
```

```
## [1] -0.5914735
```

```
FisherFun(0)
```

```
## [1] -0.5914735
```

```
FisherFun(1.5)
```

```
## [1] 3.021345
```

```
FisherFun(4)
```

```
## [1] 3.021345
```

```
FisherFun(4.7)
```

```
## [1] 3.021345
```

```
FisherFun(7)
```

```
## [1] 3.021345
```

```
FisherFun(8)
```

```
## [1] 3.021345
```

```
FisherFun(38)
```

```
## [1] 3.037648
```

```
#when FisherFun starts at -11, -1, and 0,
#it will return an optimal value at -0.5914735
#when FisherFun starts at 1.5, 4, 4.7, 7, and 8,
#it will return an optimal value at 3.021345
#when starts at 38, it will return at 3.037648
```

To do Refinement: plug -0.5914735, 3.021345, and 3.037648 into NewtonR function

```
NewtonR(-0.5914735)
```

```
## [1] -0.5914735
```

```
NewtonR(3.021345)
```

```
## [1] 3.021345
```

```
NewtonR(3.037648)
```

```
## [1] 3.021345
```

(e)Comments

Newton's method is a very efficient way to find roots by iteration, although it has some limitations.

Different starting points may result in different converge values. Not all the starting point can find it's optimal value efficiently by Newton-Raphson method.

Newton-Raphson methods: $G(x) = x - \frac{f(x)}{f'(x)}$

Fixed point method choses $\alpha_t = -\frac{1}{f'(x_t)}$, which simplifies $G(x) = x - \frac{f(x)}{f'(x)}$ as $G(x) = x + \alpha_t f(x)$. And the apply of α_t leads to a faster convergence order.

Part (c) involves using Fisher scoring to find MLE of θ first, then applies Newton-Raphson to do refinement. This method is the most stable one to use.

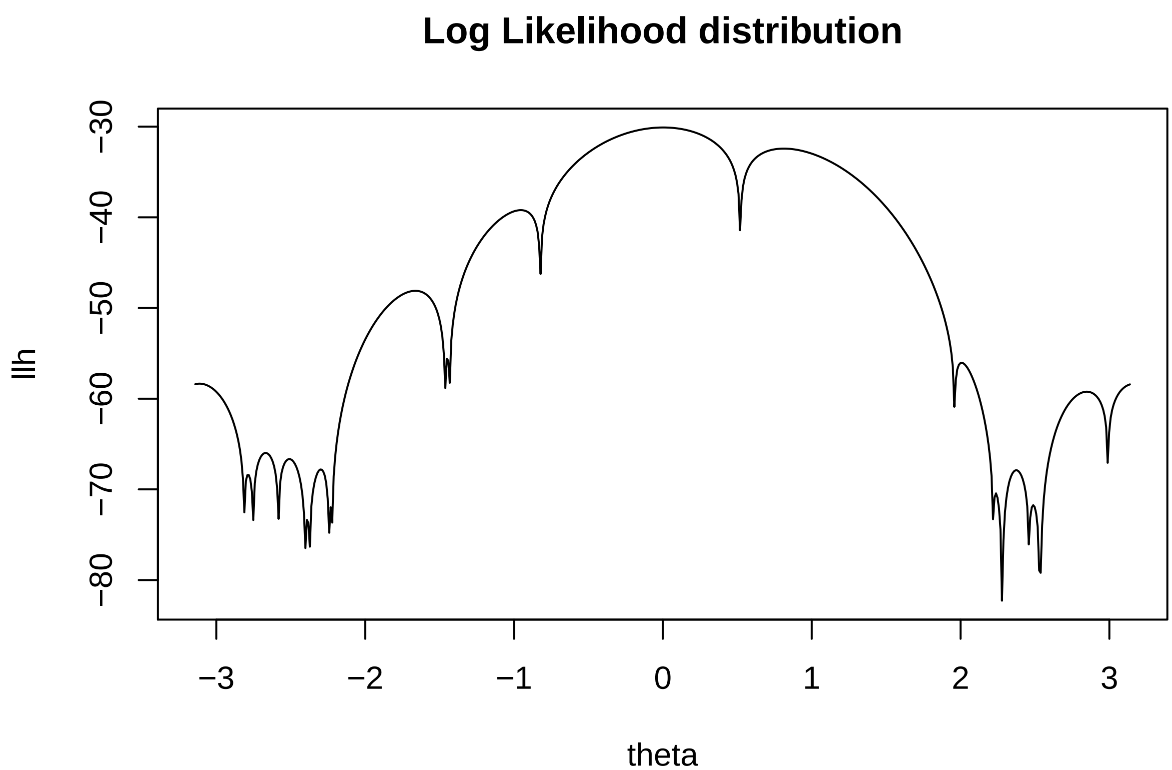
Question 2

(a) Find log-likelihood function

Given $p(x;\theta) = \frac{1-\cos(x-\theta)}{2\pi}$, do the same method as Question 1, and get:

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

```r
x2 <- c(3.91, 4.85, 2.28, 4.06, 3.70, 4.04, 5.46, 3.53, 2.28, 1.96,
       2.53, 3.88, 2.22, 3.47, 4.82, 2.46, 2.99, 2.54, 0.52)
staring_p <- c(-11,-1, 0, 1.5, 4, 4.7, 7, 8, 38)
#2(a)
theta <- seq(-pi, pi, 0.01)
n2 <- length(x2)
llh <- rep(0,length(theta))
for (i in 1:length(theta)) {
  llh[i] <- sum(log((1 - cos(x2 - theta[i])) / (2 * pi)))
}
plot(theta, llh, type = "l", ylab = "llh", xlab = "theta",
main = "Log Likelihood distribution")
```



(b)

$$E[x|\theta] = \int_0^{2\pi} \frac{1 - cos(x - \theta)}{2\pi} dx = \frac{1}{2\pi}[2\pi - \int_0^{2\pi} cos(x - \theta)]dx = 1$$

$E[x|\theta] = 1$ means that the MME for θ is just the sample mean

```r
mme <- mean(x2)
print(mme)
```

```
## [1] 3.236842
```

(c)

```r
llh1stDeriv <- function (theta){
  w <- -(sum(sin(x2 - theta)/(1 - cos(x2 - theta))))
  return(w)
}
llh2ndDeriv <- function(theta){
  k <- -(sum(1 / (1 - cos(x2 - theta))))
  return(k)
}

nr_method <- function(theta0){
  theta <- array()
  theta[1] <- theta0
  i <- 1
  difference <- 10
  while (abs(difference)>10^(-10) & i<200) {
    theta[i+1] <- theta[i] - llh1stDeriv(theta[i]) / llh2ndDeriv(theta[i])
    difference <- theta[i+1] - theta[i]
    i <- i + 1
  }
  return(theta[i])
}

nr_method(mme)
```

```
## [1] 3.170715
```

(d)

plug -2.7 and 2.7 as the initial value of θ into nr_method function

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

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

```r
nr_method(2.7)
```

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

(e)

```
new.starting.p <- seq(-pi,pi,length.out = 200)
calculation <- sapply(new.starting.p,nr_method)
converge.point <- round(calculation,6)
counting.result <- as.data.frame(table(converge.point))
print(counting.result)
```

```
##    converge.point Freq
## 1       -3.112471   11
## 2       -2.786557    2
## 3       -2.668857    5
## 4       -2.509356    6
## 5       -2.388267    1
## 6       -2.297926    4
## 7       -2.232192    1
## 8       -1.662712   24
## 9       -1.447503    1
## 10      -0.954406   19
## 11       0.003118   42
## 12       0.812637   46
## 13       2.007223    8
## 14       2.237013    2
## 15       2.374712    6
## 16        2.48845    2
## 17       2.848415   15
## 18       3.170715    5
```

Question 3

(a)

```
beetle <- 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))

pop.growth.fun <- function(t, K, r){
  f <- (2 * K) / (2 + (K - 2) * exp(-r * t))
  return(f)
}

Gauss.N <- nls(beetles~pop.growth.fun(days, K, r), data = beetle,
        start = list(K = 1300, r = 0.3), trace = TRUE)
```

```
## 2702272 :  1300.0    0.3
## 194489.2 :  884.2975241   0.1400554
## 78643.71 :  1015.9553635    0.1228027
## 73520.07 :  1045.3652874    0.1189473
## 73422.16 :  1048.9463451    0.1183845
## 73419.77 :  1049.333890    0.118288
## 73419.7 :  1049.3949623    0.1182717
## 73419.7 :  1049.405172    0.118269
## 73419.7 :  1049.4068937    0.1182685
```

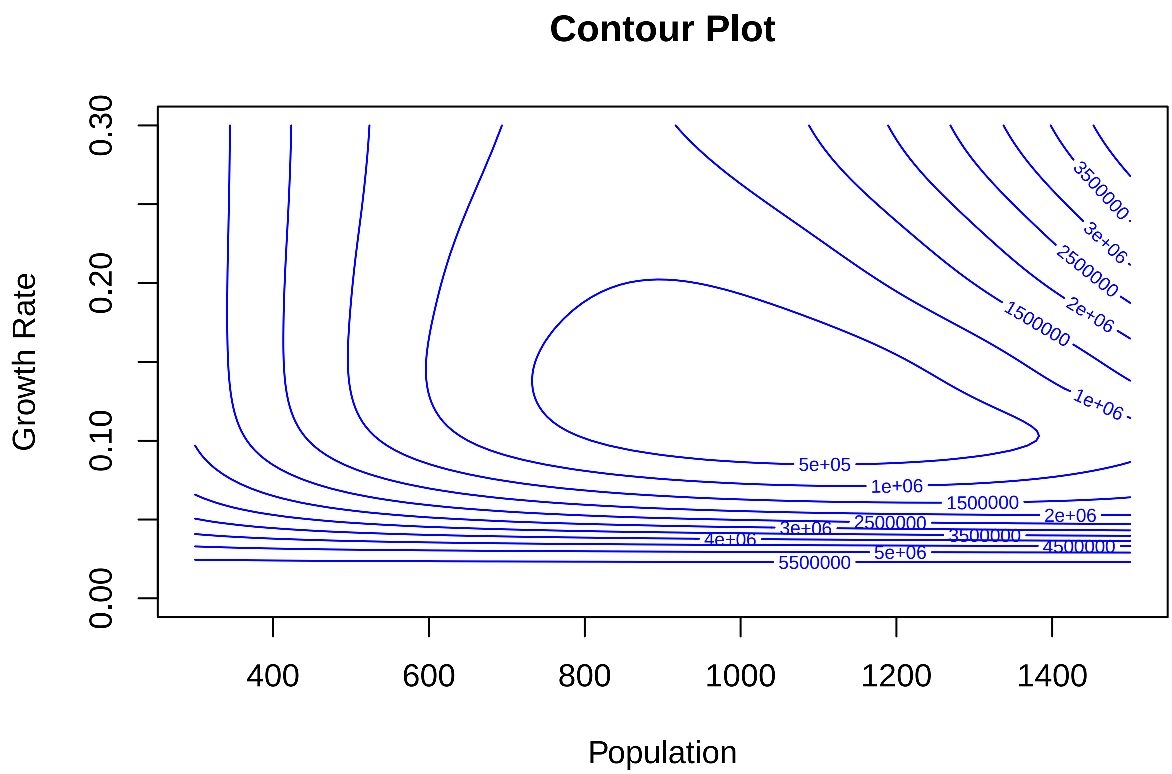
```r
summary(Gauss.N)
```

```
##
## Formula: beetles ~ pop.growth.fun(days, K, r)
##
## Parameters:
##    Estimate Std. Error t value Pr(>|t|)
## K 1.049e+03  4.717e+01   22.25 1.76e-08 ***
## r 1.183e-01  6.533e-03   18.10 8.90e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 95.8 on 8 degrees of freedom
##
## Number of iterations to convergence: 8
## Achieved convergence tolerance: 4.295e-06
```

(b)

```r
days = c(0, 8, 28, 41, 63, 69, 97, 117, 135, 154)
beetles = c(2, 47, 192, 256, 768, 896, 1120, 896, 1184, 1024)
SSE <- function(K, r){
  sse.value <- sum((beetles - (2*K/(2+(K-2)*exp(-r*days))))^2)
  return(sse.value)
}
SSE.matri <- matrix(0,100,100, byrow = TRUE)
for (i in 1:100){
  for (j in 1:100){
    K <- 300+12*i
    r <- 0+0.003*j
    SSE.matri[i, j] <- SSE(K, r)
  }
}
K <- seq(300, 1500, length.out = 100)
```

```r
r <- seq(0, 0.3, length.out = 100)
contour(K, r, SSE.matri, col = "blue", main = "Contour Plot",
xlab = "Population", ylab = 'Growth Rate')
```



(c)

I choose to use Fisher Scoring first then to use Newton-Raphson to do refinement.

Given log N_t are i.i.d. with mean $log(f(t))$ and variance σ^2.