Home Work 2

Yaqi Dai

Problem 1

(a)

$$
\begin{aligned}
L(\theta) &= \prod_{i=1}^{n} p(x_i|\theta) \\
l(\theta) &= lnL(\theta) = ln\prod_{i=1}^{n} p(x_i|\theta) \\
&= \sum_{i=1}^{n} ln(p(x_i|\theta)) \\
&= \sum_{i=1}^{n} ln\frac{1}{\pi[1+(\theta-x_i)^2]} \\
&= 0 - \sum_{i=1}^{n} ln\pi - \sum_{i=1}^{n} ln[1+(\theta-x_i)^2] \\
&= -nln\pi - \sum_{i=1}^{n} ln[1+(\theta-x_i)^2]
\end{aligned}
$$

$$
\begin{aligned}
l'(\theta) &= \frac{dl(\theta)}{d\theta} \\
&= 0 - \sum_{i=1}^{n} \frac{2(\theta-x_i)}{1+(\theta-x_i)^2} \\
&= -2\sum_{i=1}^{n} \frac{\theta-x_i}{1+(\theta-x_i)^2}
\end{aligned}
$$

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

Note $I(\theta) = nI_x(\theta)$

$$
\begin{aligned}
I_x(\theta) &= E[(\frac{\partial}{\partial\theta} log(p(x;\theta)))^2] \\
&= E[p'(x)^2] \\
&= \int_{-\infty}^{+\infty} \frac{\{p'(x)\}^2}{p(x)} \mathrm{d}x
\end{aligned}
$$

$$
\begin{aligned}
I_x(\theta) &= \int_{-\infty}^{+\infty} (\frac{2(x-\theta)}{1+(x-\theta)^2})^2 \frac{1}{\pi(1+(x-\theta)^2)} \mathrm{d}x \\
&= \frac{4}{\pi} \int_{-\infty}^{+\infty} \frac{(x-\theta)^2}{(1+(x-\theta)^2)^3} \mathrm{d}x \\
&= \frac{4}{\pi} \int_{-\infty}^{+\infty} \frac{(x-\theta)^2}{(1+(x-\theta)^2)^3} \mathrm{d}(x-\theta)
\end{aligned}
$$

Rewrite the equation:

$$
I_x(\theta) = \frac{4}{\pi} \int_{-\infty}^{+\infty} \frac{x^2}{(1+x^2)^3} \mathrm{d}x
$$

Now let $u = \frac{1}{1+x^2}$, then we can have $x = (\frac{1-u}{u})^{\frac{1}{2}}$, $\mathrm{d}x = -\frac{1}{2}(1-u)^{-\frac{1}{2}} u^{-\frac{3}{2}} \mathrm{d}u$

$$
\begin{aligned}
I_x(\theta) &= \frac{8}{\pi} \int_0^{+\infty} \frac{x^2}{(1+x^2)^3} \mathrm{d}x \\
&= \frac{8}{\pi} \int_1^0 \frac{1-u}{u} u^3 (-\frac{1}{2})(1-u)^{-\frac{1}{2}} u^{-\frac{3}{2}} \mathrm{d}u \\
&= \frac{4}{\pi} \int_0^1 (1-u)^{\frac{1}{2}} u^{\frac{1}{2}} \mathrm{d}u \\
&= \frac{4}{\pi} \int_0^1 (1-u)^{\frac{3}{2}-1} u^{\frac{3}{2}-1} \mathrm{d}u \\
&= \frac{4}{\pi} B(\frac{3}{2}, \frac{3}{2}) \\
&= \frac{4}{\pi} \frac{\Gamma(\frac{3}{2})\Gamma(\frac{3}{2})}{\Gamma(3)} \\
&= \frac{4}{\pi} \frac{\frac{1}{2}\sqrt{\pi}\frac{1}{2}\sqrt{\pi}}{2} \\
&= \frac{1}{2}
\end{aligned}
$$

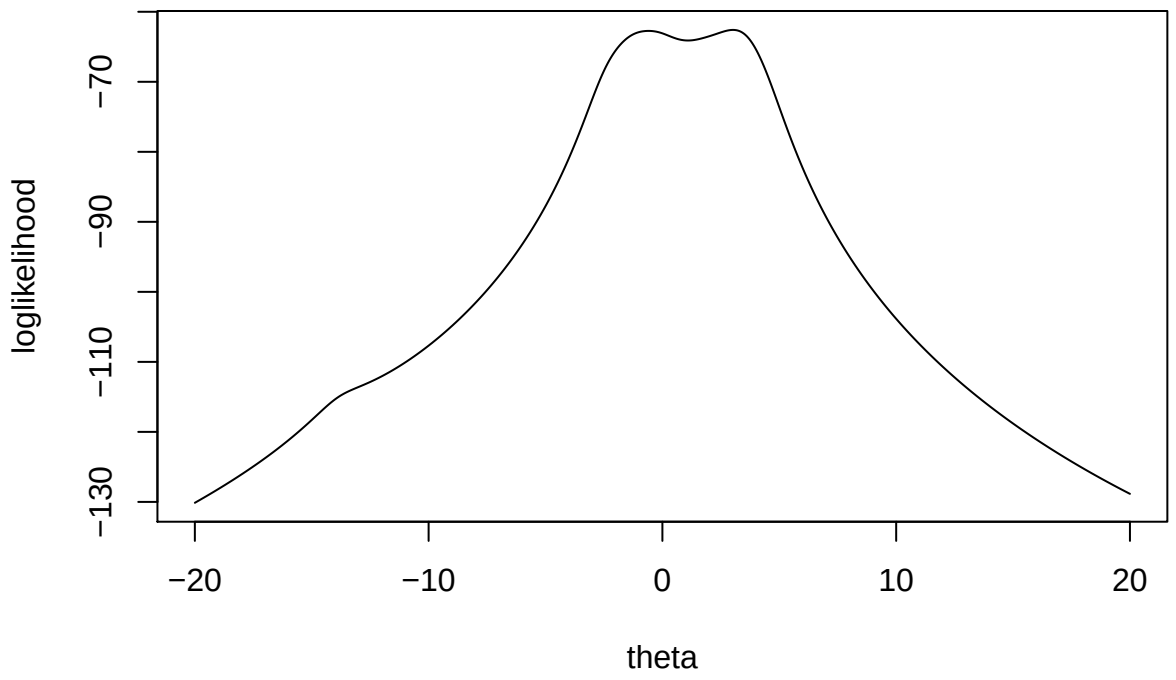
As $I(\theta) = nI_x(\theta)$, we can have

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

(b)

Graph the log-likelihood function.

```r
Loglikelihood <- function(X, theta){
  n <- length(X)
  -n*log(pi) - sum(log(1 + (theta - X)^2))
}
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)
theta <- seq(-20, 20, by = 0.01)
loglikelihood <- sapply(theta, FUN=function(theta) Loglikelihood(X, theta))
plot(theta, loglikelihood, type="l")
```



All MLE for θ using the Newton-Raphson method are in the following table.

```r
## function to calculate the MLE of Cauchy distribution by Newton-Raphson method
NewtonMleCauchy <- function(X, a, tolar){
  # X is vector of observed sample
  # a is the start value
  timestart <- Sys.time()
  n <- length(X)
  theta.iterate <- a
  i <- 0 # i is the number of iteration
  #calculate the first derivate of log likelyhood of Cauchy distribution
  firstd <- (-2)*sum((theta.iterate - X)/(1 + (theta.iterate - X)^2))
  while(abs(firstd) > tolar ){
    secondd <- (-2)*sum(((1 - (theta.iterate - X)^2)/(1 + (theta.iterate - X)^2))^2)
    theta.new <- theta.iterate - firstd/secondd
    theta.iterate <-theta.new
    firstd <- (-2)*sum((theta.iterate - X)/(1 + (theta.iterate - X)^2))
```

```
    i <- i + 1
  }
  timeend <- Sys.time()
  running.time <- timeend - timestart
  result1 <- c(theta.iterate, running.time, i)
  return(result1)
}
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)
Y <- c(-11, -1, 0, 1.5, 4, 4.7, 7, 8, 38)

## find MLE for theta
mle.matrix <- sapply(Y, FUN = function(Y) NewtonMleCauchy(X, Y, 1E-10))

## start at sample mean
sample.mean <- mean(X)
theta.sm <- NewtonMleCauchy(X, sample.mean, 1E-10)
```

Table 1: result from Netown-Raphson method

starting point	MLE	running time	number of iterations
-11.0	-0.5914735	0.0008550	312
-1.0	-0.5914735	0.0010009	232
0.0	-0.5914735	0.0000000	238
1.5	3.0213454	0.0000000	145
4.0	3.0213454	0.0000000	122
4.7	3.0213454	0.0000000	124
7.0	3.0213454	0.0000000	132
8.0	3.0213454	0.0010018	138
38.0	3.0213454	0.0010021	881

The sample mean is 3.25778, if we use sample mean as a start value, the MLE for θ is 3.0213454 and it iterates 117 times. The MLE for θ is the same as above and the number of interates is much lower. So I think the sample is a good starting point.

(c)

When apply fixed-point iteration, use the 2000 as a maximum iteration number to be a limit. The main code for the function shows below.

```
Fixedpoint <- function(alfa, a, totlar){
  # a is the initial value of theta
  timestart <- Sys.time()
  theta.iterate <- a
  firstd <- (-2)*sum((theta.iterate - X)/(1 + (theta.iterate - X)^2))
  theta.new <- theta.iterate + alfa*firstd
  i <- 0
  while (abs(firstd) > totlar && i <= 2000) {
    theta.new <- theta.iterate + alfa*firstd
    theta.iterate <- theta.new
    firstd <- (-2)*sum((theta.iterate - X)/(1 + (theta.iterate - X)^2))
    i <- i + 1
```

```
  }
  timeend <- Sys.time()
  runningtime <- timestart - timeend
  result2 <- c(theta.iterate, runningtime, i)
  return(result2)
}
```

Then we can have the following result.

Table 2: result for alpha= 1

starting point	MLE	running time	number of iterations
-11.0	-0.5914735	-0.0010390	365
-1.0	0.2417269	-0.0020030	2001
0.0	-1.1713919	-0.0020070	2001
1.5	0.2417269	-0.0025971	2001
4.0	-1.1713919	-0.0038519	2001
4.7	-1.1063091	0.0000000	2001
7.0	-1.1063091	-0.0043290	2001
8.0	0.1035079	-0.0020370	2001
38.0	0.1035079	0.0000000	2001

Table 3: result for alpha= 0.64

starting point	MLE	running time	number of iterations
-11.0	-0.5914735	0.0000000	23
-1.0	-0.5914735	0.0000000	17
0.0	-0.5914735	0.0000000	17
1.5	3.2398379	-0.0072789	2001
4.0	-0.5914735	0.0000000	18
4.7	-0.5914735	0.0000000	18
7.0	2.5915177	-0.0020051	2001
8.0	-0.5914735	0.0000000	19
38.0	2.5915177	-0.0010030	2001

Table 4: result for alpha= 0.25

starting point	MLE	running time	number of iterations
-11.0	-0.5914735	0	43
-1.0	-0.5914735	0	35
0.0	-0.5914735	0	35
1.5	3.0213454	0	14
4.0	3.0213454	0	11
4.7	3.0213454	0	11
7.0	3.0213454	0	10
8.0	3.0213454	0	12
38.0	3.0213454	0	95

(d)

Main code of Fisher scoring and refine by Newton-Raphson. I only interate 20 times by Fisher-scoring method, then refine the result by Newto-Raphson method.

```
## function to calculate the MLE of Cauchy distribution by Fisher scoring
FisherScoring <- function(X, a, tolar){
  # X is vector of observed sample
  # a is the start value
  n <- length(X)
  fisherinfor <- n/2
  theta.iterate <- a
  #calculate the first derivate of log likelyhood of Cauchy distribution
  firstd <- (-2)*sum((theta.iterate - X)/(1 + (theta.iterate - X)^2))
  i <- 0
  while(abs(firstd) > tolar && i <= 20){
    theta.new <- theta.iterate + firstd/fisherinfor
    theta.iterate <-theta.new
    firstd <- (-2)*sum((theta.iterate - X)/(1 + (theta.iterate - X)^2))
    i <- i + 1
  }
  return(theta.iterate)
}
## use theta calculated from Fisher-scoring method as the initial value
## in the Newton-Raphson method

NewtonFisher <- function(X, a, tolar){
  theta.fisher <- FisherScoring(X, a, tolar)
  theta.newton <- NewtonMleCauchy(X, theta.fisher, tolar)
  return(theta.newton)
}
```

Result from fisher-scoring and the refined result shows below.

Table 5: result from fisher-scoring and refined by Newton-Raphson method

starting point	fisher estimate	refined MLE	running time	number of iterations
-11.0	-1.4270537	-0.5914735	0.0000000	238
-1.0	-0.5937413	-0.5914735	0.0000000	180
0.0	-0.5880421	-0.5914735	0.0000000	184
1.5	3.0188665	3.0213454	0.0000000	94
4.0	3.0213484	3.0213454	0.0000000	60
4.7	3.0213498	3.0213454	0.0010021	62
7.0	3.0213664	3.0213454	0.0000000	70
8.0	3.0213984	3.0213454	0.0000000	75
38.0	36.0677379	3.0213454	0.0000000	797

(e)

Newton-Raphson method is stable for different starting points. Results from Fixed-point method rely on α. If α is good, the speed for Fixed-point method is faster than other method, it will have the least iterate steps. But if α is not good enough, in the test above, it always connot have the result within 2000 iterate steps. If

we use Fisher-scoring method to find MLE then refine the result by Newyon-Raphson method, it will need fewer steps to iterate. Actually, within in 20 iterate steps, many of the results above is really close to the refined MLE.

Problem 2

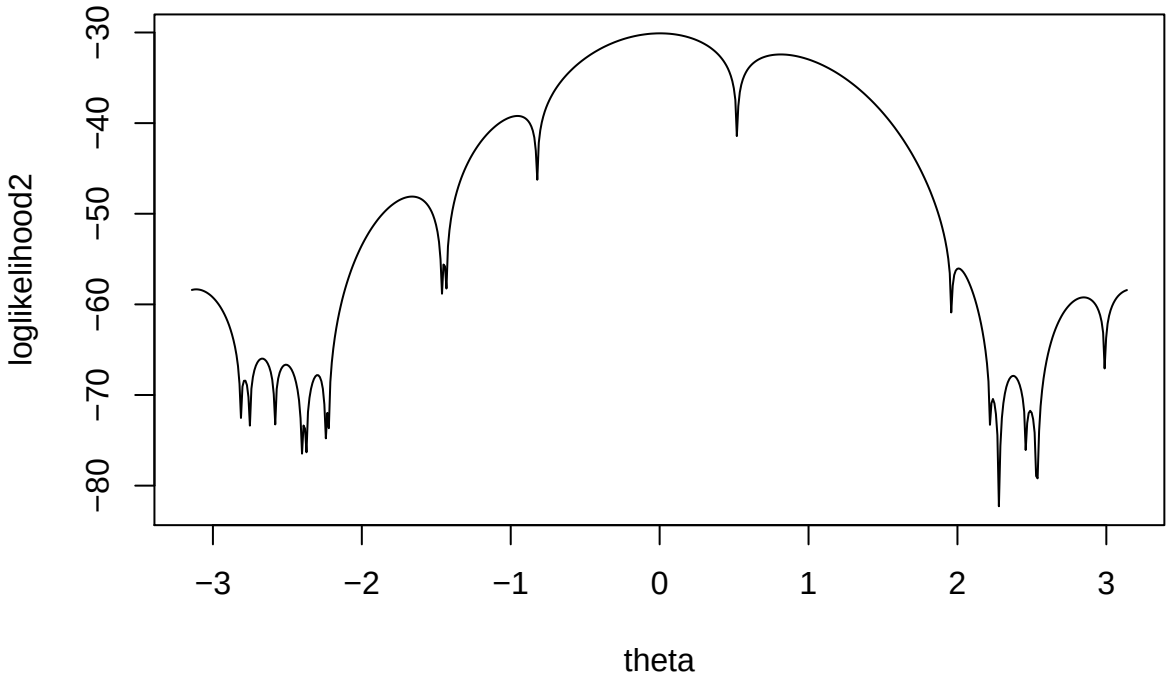
(a)

Log-likelihood functions of θ based on sample is :

$$\begin{aligned} L(\theta) &= \prod_{i=1}^{n} p(x_i|\theta) \\ l(\theta) &= \sum_{i=1}^{n} ln\frac{1-cos(x_i-\theta)}{2\pi} \\ &= \sum_{i=1}^{n}(1-cos(x_i-\theta)) - nln2\pi \end{aligned}$$

Graph the function between $-\pi$ and π

```
Loglikelihood <- function(X, theta){
  n <- length(X)
  sum(log(1 - cos(X - theta))) - n*log(2*pi)
}

X <- 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)
theta <- seq(-pi, pi, by = 0.01)
loglikelihood2 <- sapply(theta, FUN=function(theta) Loglikelihood(X, theta))
plot(theta, loglikelihood2, type="l")
```



(b)

$$
\begin{aligned}
E[X|\theta] &= \int_0^{2\pi} xp(x|\theta)\mathrm{d}x \\
&= \int_0^{2\pi} \frac{x - xcos(x-\theta)}{2\pi}\mathrm{d}x \\
&= [\frac{x^2}{4\pi} - \frac{1}{2\pi}[xsin(x-\theta) + cos(x-\theta)]]\Big|_0^{2\pi} \\
&= \pi + sin\theta
\end{aligned}
$$

$$
\overline{x} = \pi + sin\theta
$$

$$
\begin{aligned}
\theta_0 = \hat{\theta}_{moment} &= \arcsin(\overline{x} - \pi) \\
&= 0.095394067
\end{aligned}
$$

(c)

Main code for Newton-Raphson method is:

```
NewtonMethod <- function(X, a, tolar){
  # X is vector of observed sample
  # a is the start value
```

```
  n <- length(X)
  theta.iterate <- a
  #calculate the first derivate of log likelyhood of Cauchy distribution
  firstd <- sum(sin(X - theta.iterate)/(cos(X - theta.iterate) - 1))
  while(abs(firstd) > tolar ){
    secondd <- sum(1/(cos(X - theta.iterate) - 1))
    theta.new <- theta.iterate - firstd/secondd
    theta.iterate <-theta.new
    firstd <- sum(sin(X - theta.iterate)/(cos(X - theta.iterate) - 1))
  }
  return(theta.iterate)
}
```

MLE for θ using Newton-Raphson method with $\theta_0 = \hat{\theta}_{moment}$ is 0.003118157.

(d)

$\theta_0 = -2.7$, MLE for θ is -2.668857. $\theta_0 = 2.7$, MLE for θ is 2.848415.

(e)

We can use it devied the set into 18 sets.

Main code for (e) is :

```
initialvalue <- seq(-pi, pi, length.out = 200)
theta.set <- round(sapply(initialvalue,
                          FUN = function(initialvalue)
                            NewtonMethod(X, initialvalue, 1E-10)), 4)
theta.list <- split(initialvalue, theta.set)
list(theta.list)
```

```
## [[1]]
## [[1]]$`-3.1125`
##  [1] -3.141593 -3.110019 -3.078445 -3.046871 -3.015297 -2.983724 -2.952150
##  [8] -2.920576 -2.889002 -2.857428 -2.825855
##
## [[1]]$`-2.7866`
## [1] -2.794281 -2.762707
##
## [[1]]$`-2.6689`
## [1] -2.731133 -2.699560 -2.667986 -2.636412 -2.604838
##
## [[1]]$`-2.5094`
## [1] -2.573264 -2.541691 -2.510117 -2.478543 -2.446969 -2.415395
##
## [[1]]$`-2.3883`
## [1] -2.383822
##
## [[1]]$`-2.2979`
## [1] -2.352248 -2.320674 -2.289100 -2.257526
##
## [[1]]$`-2.2322`
```

```
## [1] -2.225953
##
## [[1]]$`-1.6627`
##  [1] -2.194379 -2.162805 -2.131231 -2.099657 -2.068084 -2.036510 -2.004936
##  [8] -1.973362 -1.941788 -1.910215 -1.878641 -1.847067 -1.815493 -1.783919
## [15] -1.752346 -1.720772 -1.689198 -1.657624 -1.626050 -1.594477 -1.562903
## [22] -1.531329 -1.499755 -1.468181
##
## [[1]]$`-1.4475`
## [1] -1.436608
##
## [[1]]$`-0.9544`
##  [1] -1.4050339 -1.3734601 -1.3418863 -1.3103125 -1.2787387 -1.2471649
##  [7] -1.2155911 -1.1840173 -1.1524435 -1.1208697 -1.0892959 -1.0577221
## [13] -1.0261484 -0.9945746 -0.9630008 -0.9314270 -0.8998532 -0.8682794
## [19] -0.8367056
##
## [[1]]$`0.0031`
##  [1] -0.80513179 -0.77355799 -0.74198419 -0.71041040 -0.67883660
##  [6] -0.64726281 -0.61568901 -0.58411522 -0.55254142 -0.52096763
## [11] -0.48939383 -0.45782003 -0.42624624 -0.39467244 -0.36309865
## [16] -0.33152485 -0.29995106 -0.26837726 -0.23680347 -0.20522967
## [21] -0.17365588 -0.14208208 -0.11050828 -0.07893449 -0.04736069
## [26] -0.01578690  0.01578690  0.04736069  0.07893449  0.11050828
## [31]  0.14208208  0.17365588  0.20522967  0.23680347  0.26837726
## [36]  0.29995106  0.33152485  0.36309865  0.39467244  0.42624624
## [41]  0.45782003  0.48939383
##
## [[1]]$`0.8126`
##  [1] 0.5209676 0.5525414 0.5841152 0.6156890 0.6472628 0.6788366 0.7104104
##  [8] 0.7419842 0.7735580 0.8051318 0.8367056 0.8682794 0.8998532 0.9314270
## [15] 0.9630008 0.9945746 1.0261484 1.0577221 1.0892959 1.1208697 1.1524435
## [22] 1.1840173 1.2155911 1.2471649 1.2787387 1.3103125 1.3418863 1.3734601
## [29] 1.4050339 1.4366077 1.4681815 1.4997553 1.5313291 1.5629029 1.5944767
## [36] 1.6260505 1.6576243 1.6891981 1.7207719 1.7523457 1.7839194 1.8154932
## [43] 1.8470670 1.8786408 1.9102146 1.9417884
##
## [[1]]$`2.0072`
## [1] 1.973362 2.004936 2.036510 2.068084 2.099657 2.131231 2.162805 2.194379
##
## [[1]]$`2.237`
## [1] 2.225953 2.257526
##
## [[1]]$`2.3747`
## [1] 2.289100 2.320674 2.352248 2.383822 2.415395 2.446969
##
## [[1]]$`2.4884`
## [1] 2.478543 2.510117
##
## [[1]]$`2.8484`
##  [1] 2.541691 2.573264 2.604838 2.636412 2.667986 2.699560 2.731133
##  [8] 2.762707 2.794281 2.825855 2.857428 2.889002 2.920576 2.952150
## [15] 2.983724
##
```

```
## [[1]]$`3.1707`
## [1] 3.015297 3.046871 3.078445 3.110019 3.141593
```

problem 3

(a)

Main code for fit the population growth shows below:

```
beetles <- data.frame(
  days <- c(0, 8, 28, 41, 63, 69, 97, 117, 135, 154),
  nbeetles <- c(2, 47, 192, 256, 768, 896, 1120, 896, 1184, 1024)
)
K <- 1500
r1 <- log(beetles$nbeetles*(K - 2)/(K - beetles$nbeetles)*2)
r2 <- r1 / beetles$days....c.0..8..28..41..63..69..97..117..135..154.
r.mean <- mean(r2[2: 10])
N0 <- beetles[1,2]

nls(nbeetles ~ (K*N0)/(N0 + (K - N0)*exp(-r*days)),
    start = list(K = 1500, r = r.mean), data = beetles, trace = TRUE)
```

```
## 1980050 :  1500.0000000    0.1652033
## 117991 :  997.5137694   0.1394446
## 78622.18 :  1019.990949    0.123586
## 73557.02 :  1044.6290612    0.1190586
## 73423.02 :  1048.865917    0.118403
## 73419.79 :  1049.3219772    0.1182912
## 73419.7 :  1049.3929896    0.1182723
## 73419.7 :  1049.4048387    0.1182691
## 73419.7 :  1049.4068372    0.1182685
```

```
## Nonlinear regression model
##   model: nbeetles ~ (K * N0)/(N0 + (K - N0) * exp(-r * days))
##    data: beetles
##         K         r
## 1049.4068    0.1183
##  residual sum-of-squares: 73420
##
## Number of iterations to convergence: 8
## Achieved convergence tolerance: 4.988e-06
```

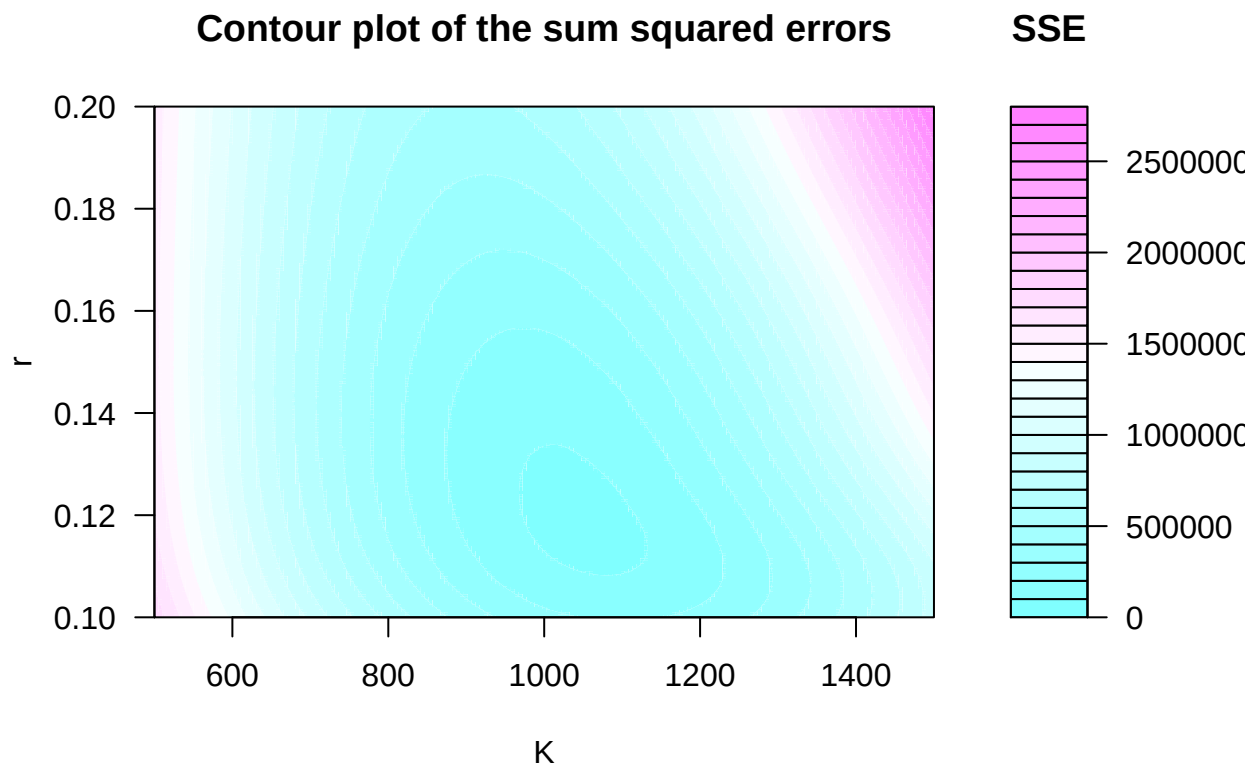
We can the optimized value of K and r to be 1049.4068 and 0.1183

(b)

Counter plot of the sum of squared errors(SSE) is:

```
K1 <- seq(500, 1500, length.out = 200)
r1 <- seq(0.1, 0.2, length.out = 200)

n.beetles <- as.numeric(beetles$nbeetles....c.2..47..192..256..768..896..1120..896..1184..1024.)
n.days <- as.numeric(beetles$days....c.0..8..28..41..63..69..97..117..135..154.)
```

```r
SSE <- function(K1, r1){
  error.sq <- sum((n.beetles - (N0*K1)/(N0 + (K1 - N0)*exp(-r1*n.days)))^2)
  return(error.sq)
}

matrix.a <- matrix(nrow = 200, ncol = 200)

for(i in 1:200){
  for(j in 1:200){
    matrix.a[i, j] <- SSE(K1[i], r1[j])
  }
}

filled.contour(K1, r1, matrix.a, plot.title = title(main = "Contour plot of the sum squared errors",
                                                    xlab = "K", ylab = "r"),
               key.title = title(main = "SSE"))
```



(c)

```r
nloglikelihood.c <- function(theta, N, days){
  K <- theta[1]
  r <- theta[2]
  sigma <- theta[3]
  t <- days
```

```
  mu <- log((K*N0)/(2 + (K - N0)*exp(-r*t)))
  theta.iterate <- (- sum(dnorm(log(N), mu, sigma, log = TRUE)))
}


sq <- sqrt(var(log(beetles$nbeetles)))

theta0 <- c(K, r.mean, sq)
mle <- nlm(nloglikelihood.c, theta0, N = beetles$nbeetles,days = beetles$days,hessian = TRUE)
estimate <- mle$estimate
estimate
```

```
## [1] 820.3814175   0.1926394   0.6440835
```

```
matrix.hessian <- mle$hessian
matrix.hessian
```

```
##               [,1]         [,2]          [,3]
## [1,]  2.389708e-05   0.05442391 -3.029013e-06
## [2,]  5.442391e-02 373.52687023 -5.759091e-02
## [3,] -3.029013e-06  -0.05759091  4.817369e+01
```

```
variance <- solve(matrix.hessian)
variance
```

```
##               [,1]          [,2]          [,3]
## [1,]  6.262775e+04 -9.125039e+00 -6.971005e-03
## [2,] -9.125039e+00  4.006728e-03  4.216227e-06
## [3,] -6.971005e-03  4.216227e-06  2.075822e-02
```

```
diag(variance)
```

```
## [1] 6.262775e+04 4.006728e-03 2.075822e-02
```

The estimate values are

$$(K, r, \sigma) = (820.3814175, 0.1926394, 0.6440835)$$

and the varance of the parameters are

$$(var(K), var(r), var(\sigma)) = (6.262775 * 10^4, 4.006728 * 10^{-3}, 2.075822 * 10^{-2})$$