Homework 4 - STAT 5362 Statistical Computing

Sen Yang[*]

29 September 2018

Abstract

This homework is made of Excercise 3.3.2 - Many local maxima and Excercise 3.3.3 - Modeling beetle data.

Contents

[*]sen.2.yang@uconn.edu; M.S. student at Department of Statistics, University of Connecticut.

1 Many local maxima

1.1 The log-likelihood function and corresponding plot of θ

The probability density function with parameter θ is

$$f(x;\theta) = \frac{1-\cos(x-\theta)}{2\pi}, 0 \leq x \leq 2\pi, \theta \in (-\pi, \pi).$$

Then, the likelihood function is

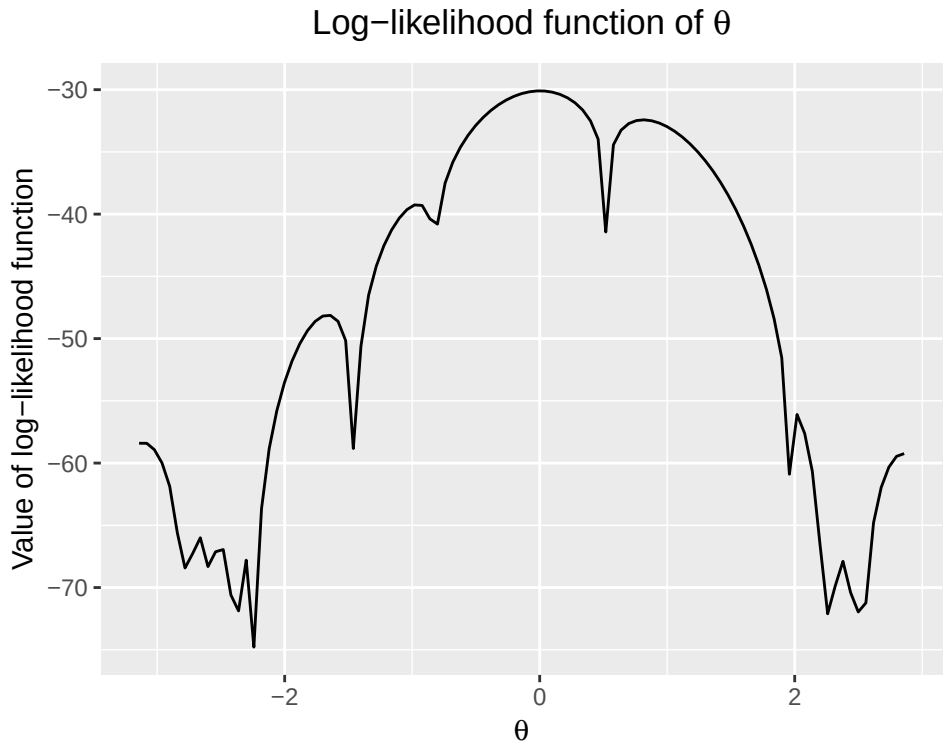
$$L(x;\theta) = (2\pi)^{-n} \prod_{i=1}^{n} [1-\cos(X_i - \theta)].$$

Take logarithm on both sides,

$$\ell(x;\theta) = \sum_{i=1}^{n} \log[1-\cos(X_i-\theta)] - n\log 2\pi.$$

With a random sample from the distribution, we calculate the value of log-likelihood function , and then plot it with $\theta \in (-\pi, \pi)$.

```
## Log-likelihood function
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)
log_llh <- function (theta, sample_X) {
  log_llh <- 0
  for (i in 1:length(sample_X)) {
  log_llh <- log_llh + log(1-cos(sample_X[i] - theta)) - log(2*pi)
  }
  log_llh
}
## plot
library(ggplot2)
ggplot() + stat_function(aes(-pi:pi), fun = log_llh, args = list(sample_X = x)) +
  labs(title = expression(paste("Log-likelihood function of ", theta)),
       x = expression(theta), y = "Value of log-likelihood function") +
  theme(plot.title = element_text(hjust = 0.5))
```



1.2 Method-of-moments estimator of θ

The expectation of X is

$$\begin{aligned}
\mathbb{E}(x|\theta) &= \int_0^{2\pi} x f(x) dx \\
&= \int_0^{2\pi} \frac{x[1-\cos(x-\theta)]}{2\pi} dx \\
&= \frac{1}{2\pi} \int_0^{2\pi} [x - x\cos(x-\theta)] dx \\
&= \frac{1}{2\pi} [2\pi^2 - \int_0^{2\pi} x \sin(x-\theta) dx] \\
&= \frac{1}{2\pi} [2\pi^2 + 2\pi \sin\theta - \int_0^{2\pi} \sin(x-\theta) dx] \\
&= \frac{1}{2\pi} [2\pi^2 + 2\pi \sin\theta] \\
&= \pi + \sin\theta.
\end{aligned}$$

Let $\mathbb{E}(x|\theta) = \bar{X_n}$, then

$$\tilde{\theta_n} = \arcsin(\bar{X_n} - \pi).$$

Calculated by R,

```r
## MOM estimation of theta
theta_mom <- asin(mean(x) - pi)
theta_mom
```

```
## [1] 0.09539407
```

1.3 Find the MLE for θ using the Newton–Raphson method

With initial value $\theta_0 = \tilde{\theta_n}$,

```r
## MLE for theta with initial value being estimation of theta by MOM
### First derivative of loglikelihood
first_derv <- function(theta, sample_X) {
  first_derv <- 0
  for (i in 1:length(sample_X)){
    first_derv <- first_derv -
      (sin(sample_X[i] - theta)/(1 - cos(sample_X[i] - theta)))
  }
  first_derv
}

### Second derivative of loglikelihood
second_derv <- function(theta, sample_X) {
  second_derv <- 0
  for (i in 1:length(sample_X)){
    second_derv <- second_derv + (1/(cos(sample_X[i] - theta) - 1))
  }
  second_derv
}

### Newton-Raphson method
newton <- function(init, pre=.Machine$double.neg.eps, maxrun=200) {
  n <- 1
  xt <- init
  while (n < maxrun){
    fx <- first_derv(xt, x)
    fx_d <- second_derv(xt, x)
    if (fx == 0) {break}
    ht <- -fx/fx_d
    xt1 <- xt + ht
    if (abs(xt1-xt) < pre) {break}
    xt <- xt1
    n <- n+1
  }
  return(c(initial = init, root = xt, iter = n))
}
```

```r
newton(theta_mom)
```

```
##     initial        root        iter
## 0.095394067 0.003118157 5.000000000
```

1.4 Newton–Raphson method with initial value $\theta_0 = -2.7$ and $\theta_0 = 2.7$

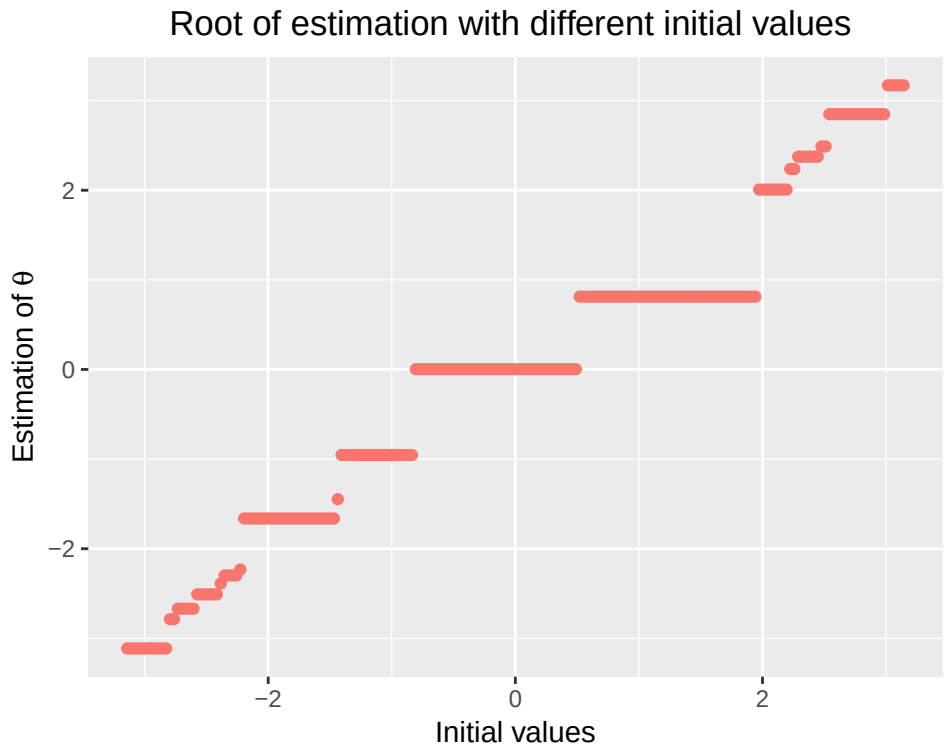
```r
result2 <- as.data.frame(matrix(0,2,3))
init2 <- c(-2.7, 2.7)
for (i in 1:length(init2)) {
  result2[i,] <- newton(init2[i])
}
colnames(result2) <- c("Initial value", "Root", "Iteration #")
library(pander)
pander(result2, caption = "The result of Newton-Raphson method optimization")
```

Table 1: The result of Newton-Raphson method optimization

Initial value	Root	Iteration #
-2.7	-2.669	7
2.7	2.848	7

1.5 Repeat the above using 200 equally spaced starting values between π and π

```r
## 200 starting values
init3 <- seq(-pi, pi, length.out = 200)
result3 <- as.data.frame(matrix(0,200,4))
for (i in 1:length(init3)) {
  result3[i,2:4] <- newton(init3[i])
  result3[i,1] <- i
}
colnames(result3) <- c("#", "Initial value", "Root", "Iteration #")
ggplot(result3, aes(result3[,2],result3[,3])) + geom_point(aes(col = "r")) +
  labs(title = "Root of estimation with different initial values ",
       x = "Initial values", y = expression(paste("Estimation of ", theta))) +
  theme(plot.title = element_text(hjust = 0.5), legend.position="none")
```




```
### Group
group_root <- result3
group_root[,3] <- round(group_root[,3],digits = 5)
library(gsubfn)
library(proto)
library(RSQLite)
library(sqldf)
group_root <- sqldf(
  'SELECT min([#]), [Initial value], Root, [Iteration #]
  FROM group_root
  GROUP BY Root'
  )
for (i in 1: dim(group_root)[1]) {
  if (i == dim(group_root)[1]) {
    group_root[i,1] <- paste(group_root[i,1], " - 200")
  } else {
    group_root[i,1] <- paste(group_root[i,1], " - ", as.numeric(group_root[i+1,1])-1)
    }
}
group_root <- group_root[,c(1,3)]
colnames(group_root) <- c("# Init", "Root")
pander(group_root, caption = "Groups with unique outcome of optimization")
```

Table 2: Groups with unique outcome of optimization

# Init	Root
1 - 11	-3.112
12 - 13	-2.787
14 - 18	-2.669
19 - 24	-2.509
25 - 25	-2.388
26 - 29	-2.298
30 - 30	-2.232
31 - 54	-1.663
55 - 55	-1.448
56 - 74	-0.9544
75 - 116	0.00312
117 - 162	0.8126
163 - 170	2.007
171 - 172	2.237
173 - 178	2.375
179 - 180	2.488
181 - 195	2.848
196 - 200	3.171

2 Modeling beetle data

2.1 Fit the population growth model to the beetles data using the Gauss-Newton approach

```
## Gauss_Newton
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[which(beetles[,1]==0),2]
fomula1 <- beetles ~ K*N0/(N0+(K-N0)*exp(-r*days))
gauss_newton <- nls(fomula1, data = beetles, start = list(K=1184, r=0.5), trace = T)
```

```
## 2215556 :  1184.0    0.5
## 981043.3 :  792.2870532   0.4131399
## 745829.4 :  794.3704521   0.2452427
## 179670.3 :  895.6172869   0.1395851
## 78418.79 :  1016.9710066    0.1227756
## 73518.6 :  1045.410578    0.118944
## 73422.14 :  1048.9489919    0.1183838
## 73419.77 :  1049.3342836    0.1182879
## 73419.7 :  1049.3950282    0.1182717
```

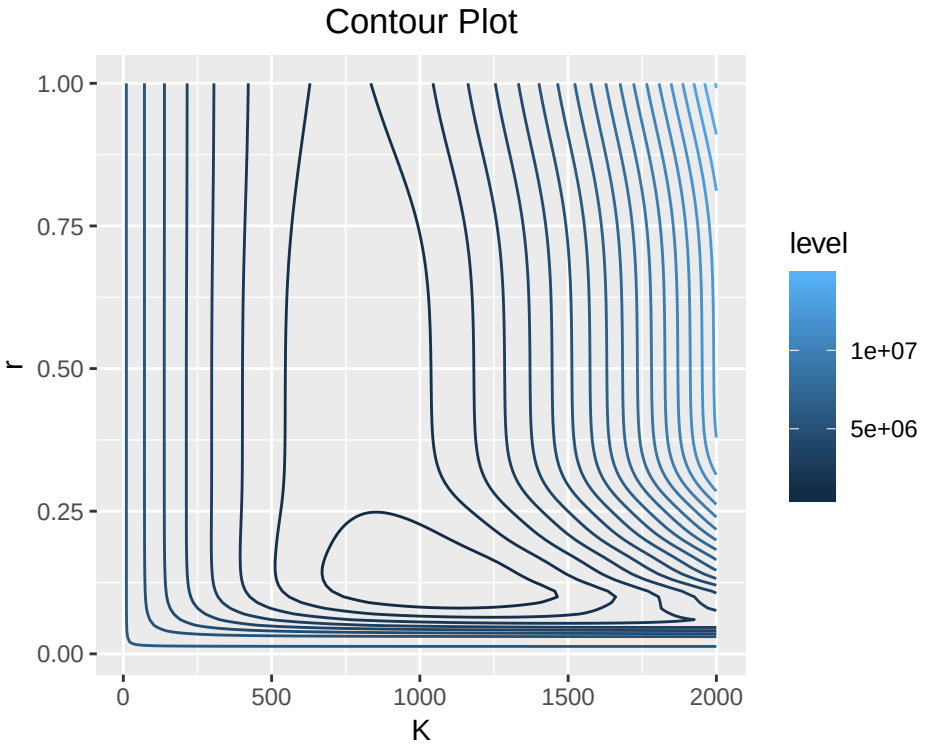
```
## 73419.7 :  1049.405183    0.118269
## 73419.7 :  1049.4068948    0.1182685
```

```
gauss_newton
```

```
## Nonlinear regression model
##   model: beetles ~ K * N0/(N0 + (K - N0) * exp(-r * days))
##    data: beetles
##         K         r
## 1049.4069    0.1183
##  residual sum-of-squares: 73420
##
## Number of iterations to convergence: 10
## Achieved convergence tolerance: 4.279e-06
```

2.2 Contour plot of the sum of squared errors

```
## Contour
### Squared Error
sqr_error <- function(r, K, sample = beetles) {
  sqr_error <- 0
  for (i in 1:dim(beetles)[1]) {
    sqr_error <- sqr_error + (K*N0/(N0+(K-N0)*exp(-r*sample[i,1]))-sample[i,2])^2
  }
  sqr_error
}
## Contour plot
K <- seq(10, 2000, 10)
r <- seq(0.01, 1, 0.01)
plot_data <- as.data.frame(matrix(0,(length(K)*length(r)),3))
colnames(plot_data) <- c("K", "r", "Squared Error")
for (i in 1:length(K)) {
  for (j in 1:length(r)) {
    plot_data[j+(i-1)*length(r),1] <- K[i]
    plot_data[j+(i-1)*length(r),2] <- r[j]
    plot_data[j+(i-1)*length(r),3] <- sqr_error(r[j], K[i])
  }
}
ggplot(plot_data, aes(x = K, y = r)) +
  geom_contour(aes(z=plot_data[,3], col = ..level..), bins = 20) +
  labs(title = "Contour Plot") + theme(plot.title = element_text(hjust = 0.5))
```



2.3 BFGS optimization with lognormility assumption

We assume that $\log N_t \overset{independent}{\sim} N(\log f(t), \sigma^2)$. The probability density function with parameter $\theta = (r, K, \sigma^2)$ is

$$g(N_t; \theta) = \frac{1}{N_t \sigma \sqrt{2\pi}} \exp(-\frac{(\log N_t - \log f(t))^2}{2\sigma^2}).$$

Then, the likelihood function is

$$L(N_i; \theta) = \prod_{i=1}^{n} \frac{1}{N_i \sigma \sqrt{2\pi}} \exp(-\frac{(\log N_i - \log f(t_i))^2}{2\sigma^2}).$$

Take logarithm on both sides,

$$\ell(x; \theta) = \sum_{i=1}^{n} log[\frac{1}{N_i \sigma \sqrt{2\pi}} \exp(-\frac{(\log N_i - \log f(t_i))^2}{2\sigma^2})]$$

$$where\ f(t_i) = \frac{KN_0}{N_0 + (K - N_0)exp(-rt_i)}.$$

Now, we get the log-likelihood function of N_t. Our objective is to maximize the log-likelihood functon given the value of parameter $\theta = (r, K, \sigma^2)$. Here, we choose to use method *BFGS*, which is a Quasi-Newton method.

```r
## BFGS
log_llh2 <- function (theta,   sample = beetles) {
  r <- theta[1]
  K <- theta[2]
  sigma_sqr <- theta[3]
  log_llh2 <- 0
  for (i in 1: dim(sample)[1]) {
    log_llh2 <- - (log_llh2 + log(1/(sample[i,2]*(sigma_sqr*2*pi)^0.5)) -
      (log(sample[i,2])-log(K*N0/(N0+(K-N0)*exp(-r*sample[i,1]))))^2/(2*sigma_sqr))
  }
  log_llh2
}
result4 <- optim(c(0.2, 1200, 100), fn = log_llh2, method = "BFGS",hessian = T)
result4
```

```
## $par
## [1]    0.4054415 1199.9999913  100.0000366
##
## $value
## [1] 3.225746
##
## $counts
## function gradient
##        8        4
##
## $convergence
## [1] 0
##
## $message
## NULL
##
## $hessian
##               [,1]          [,2]          [,3]
## [1,]  6.853647e-01 -1.904699e-06 -5.240253e-08
## [2,] -1.904699e-06  8.881784e-10  3.774758e-09
## [3,] -5.240253e-08  3.774758e-09 -9.534595e-07
```

```r
var<- diag(solve(-result4$hessian))
var
```

```
## [1] -1.467681e+00 -1.113799e+09  1.031356e+06
```