Many Local Maxima and Modeling Beetle Data

HW 4 of STAT 5361 Statistical Computing

Biju Wang[*]

09/26/2018

1 Many Local Maxima

1.1

The likelihood function is

$$L(\theta) = \prod_{i=1}^{n} \frac{1 - \cos(X_i - \theta)}{2\pi}$$

Thus, the log likelihood fucntion is

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

The following plot is the curve of log likelihood function

```
sample <- 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_sum <- function(x, sample){
  log_sum <- 0
  for (i in 1:length(sample)) {
    log_sum <- log_sum + log(1 - cos(sample[i] - x)) - log(2 * pi)
  }
  log_sum
}
library("ggplot2")
ggplot(data.frame(x = c(-pi, pi)), aes(x = x)) +
stat_function(fun = function(x) log_sum(x, sample)) +
labs(x = expression("Values of"~theta), y = expression("Log Likelihood Function"~l(theta))) +
theme(plot.title = element_text(hjust = 0.5)) +
ggtitle(expression("Log Likelihood Function vs."~theta))
```

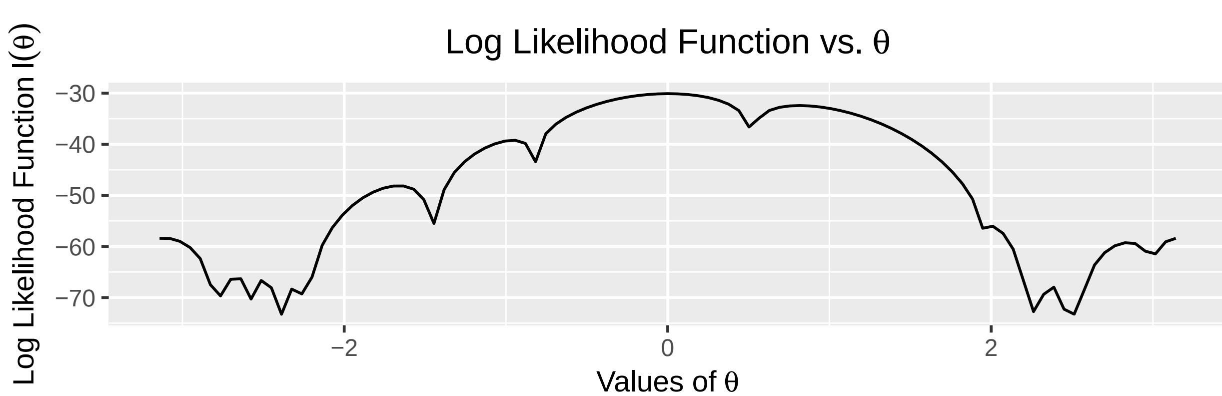

Figure 1: Log Likelihood Function vs. θ

[*]bijuwang@uconn.edu

1.2

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

Let $E(X|\theta)$ equal sample mean $\bar{X}_n$

$$\pi + \sin\theta = \bar{X}_n$$

we could obtain the method-of-moments estimator of θ is $\tilde{\theta}_n = 0.095$ or 3.046.

1.3

The first derivative is

$$l'(\theta) = \sum_{i=1}^{n} \frac{-\sin(X_i - \theta)}{1-\cos(X_i-\theta)}$$

The second derivative is

$$l''(\theta) = \sum_{i=1}^{n} \frac{1}{\cos(X_i-\theta) - 1}$$

```r
sample <- 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)

dev1_log_sum <- function(x){
  dev1_log_sum <- 0
  for (i in 1:length(sample)) {
    dev1_log_sum <- dev1_log_sum - sin(sample[i] - x)/(1 - cos(sample[i] - x))
  }
  dev1_log_sum
}

dev2_log_sum <- function(x){
  dev2_log_sum <- 0
  for (i in 1:length(sample)) {
    dev2_log_sum <- dev2_log_sum + 1/(cos(sample[i] - x) - 1)
  }
  dev2_log_sum
}

newton.raphson <- function(init, fun, fun.dev, maxiter = 100, tol = .Machine$double.eps^0.2){
  x <- init
  for (i in 1:maxiter) {
    x1 <- x -fun(x)/fun.dev(x)
    if(abs(x1 - x) < tol) break
    x <- x1
  }
  if(i == maxiter)
```

```r
    message("Reached the maximum iteration!")

  return(data.frame(root = x1, iter = i))
}

init <- c(asin(mean(sample) - pi), pi - asin(mean(sample) - pi))
res <- data.frame(init = init, root = rep(NA, length(init)))
for (i in 1:length(init)) {
  res$root[i] <- newton.raphson(init[i], dev1_log_sum, dev2_log_sum)$root
}

res_trans <- t(as.matrix(res))
rownames(res_trans) <- c("Initial Values", "Roots")

knitr::kable(res_trans, booktabs = TRUE,
             caption = "Initial Values and Roots")
```

Table 1: Initial Values and Roots

Initial Values	0.0953941	3.046199
Roots	0.0031182	3.170715

1.4

The outcomes are in the following table

```r
init <- c(-2.7, 2.7)
res <- data.frame(init = init, root = rep(NA, length(init)))
for (i in 1:length(init)) {
  res$root[i] <- newton.raphson(init[i], dev1_log_sum, dev2_log_sum)$root
}

res_trans <- t(as.matrix(res))
rownames(res_trans) <- c("Initial Values", "Roots")

knitr::kable(res_trans, booktabs = TRUE,
             caption = "Initial Values and Roots")
```

Table 2: Initial Values and Roots

Initial Values	-2.700000	2.700000
Roots	-2.668858	2.848415

1.5

```r
options(digits = 6)
init <- seq(-pi, pi, length = 200)
res <- data.frame(init = init, root = rep(NA, length(init)))
for (i in 1:length(init)) {
  res$root[i] <- newton.raphson(init[i], dev1_log_sum, dev2_log_sum)$root
}
```

```r
res <- round(res, 5)
res_trans <- t(as.matrix(res))
rownames(res_trans) <- c("Initial Values", "Roots")

library("pander")
library("ggplot2")
pander(res_trans, split.table = 120, style = 'rmarkdown', caption = "Initial Values and Roots")
```

Table 3: Initial Values and Roots (continued below)

Initial Values	-3.142	-3.11	-3.078	-3.047	-3.015	-2.984	-2.952	-2.921	-2.889
Roots	-3.112	-3.112	-3.112	-3.112	-3.112	-3.112	-3.112	-3.112	-3.112

Table 4: Table continues below

Initial Values	-2.857	-2.826	-2.794	-2.763	-2.731	-2.7	-2.668	-2.636	-2.605
Roots	-3.112	-3.112	-2.787	-2.787	-2.669	-2.669	-2.669	-2.669	-2.669

Table 5: Table continues below

Initial Values	-2.573	-2.542	-2.51	-2.479	-2.447	-2.415	-2.384	-2.352	-2.321
Roots	-2.509	-2.509	-2.509	-2.509	-2.509	-2.509	-2.388	-2.298	-2.298

Table 6: Table continues below

Initial Values	-2.289	-2.258	-2.226	-2.194	-2.163	-2.131	-2.1	-2.068	-2.037
Roots	-2.298	-2.298	-2.232	-1.663	-1.663	-1.663	-1.663	-1.663	-1.663

Table 7: Table continues below

Initial Values	-2.005	-1.973	-1.942	-1.91	-1.879	-1.847	-1.815	-1.784	-1.752
Roots	-1.663	-1.663	-1.663	-1.663	-1.663	-1.663	-1.663	-1.663	-1.663

Table 8: Table continues below

Initial Values	-1.721	-1.689	-1.658	-1.626	-1.594	-1.563	-1.531	-1.5	-1.468
Roots	-1.663	-1.663	-1.663	-1.663	-1.663	-1.663	-1.663	-1.663	-1.663

Table 9: Table continues below

Initial Values	-1.437	-1.405	-1.373	-1.342	-1.31	-1.279	-1.247	-1.216	-1.184
Roots	-1.448	-0.9544	-0.9544	-0.9544	-0.9544	-0.9544	-0.9544	-0.9544	-0.9544

Table 10: Table continues below

Initial Values	-1.152	-1.121	-1.089	-1.058	-1.026	-0.9946	-0.963	-0.9314	-0.8999
Roots	-0.9544	-0.9544	-0.9544	-0.9544	-0.9544	-0.9544	-0.9544	-0.9544	-0.9544

Table 11: Table continues below

Initial Values	-0.8683	-0.8367	-0.8051	-0.7736	-0.742	-0.7104	-0.6788	-0.6473	-0.6157
Roots	-0.9544	-0.9544	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312

Table 12: Table continues below

Initial Values	-0.5841	-0.5525	-0.521	-0.4894	-0.4578	-0.4263	-0.3947	-0.3631	-0.3315
Roots	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312

Table 13: Table continues below

Initial Values	-0.2999	-0.2684	-0.2368	-0.2052	-0.1737	-0.1421	-0.1105	-0.07893
Roots	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312

Table 14: Table continues below

Initial Values	-0.04736	-0.01579	0.01579	0.04736	0.07893	0.1105	0.1421	0.1737
Roots	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312

Table 15: Table continues below

Initial Values	0.2052	0.2368	0.2684	0.2999	0.3315	0.3631	0.3947	0.4263	0.4578
Roots	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312	0.00312

Table 16: Table continues below

Initial Values	0.4894	0.521	0.5525	0.5841	0.6157	0.6473	0.6788	0.7104	0.742
Roots	0.00312	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126

Table 17: Table continues below

Initial Values	0.7736	0.8051	0.8367	0.8683	0.8999	0.9314	0.963	0.9946	1.026
Roots	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126

Table 18: Table continues below

Initial Values	1.058	1.089	1.121	1.152	1.184	1.216	1.247	1.279	1.31
Roots	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126

Table 19: Table continues below

Initial Values	1.342	1.373	1.405	1.437	1.468	1.5	1.531	1.563	1.594
Roots	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126

Table 20: Table continues below

Initial Values	1.626	1.658	1.689	1.721	1.752	1.784	1.815	1.847	1.879
Roots	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126	0.8126

Table 21: Table continues below

Initial Values	1.91	1.942	1.973	2.005	2.037	2.068	2.1	2.131	2.163	2.194
Roots	0.8126	0.8126	2.007	2.007	2.007	2.007	2.007	2.007	2.007	2.007

Table 22: Table continues below

Initial Values	2.226	2.258	2.289	2.321	2.352	2.384	2.415	2.447	2.479	2.51	2.542
Roots	2.237	2.237	2.375	2.375	2.375	2.375	2.375	2.375	2.488	2.488	2.848

Table 23: Table continues below

Initial Values	2.573	2.605	2.636	2.668	2.7	2.731	2.763	2.794	2.826	2.857	2.889
Roots	2.848	2.848	2.848	2.848	2.848	2.848	2.848	2.848	2.848	2.848	2.848

Initial Values	2.921	2.952	2.984	3.015	3.047	3.078	3.11	3.142
Roots	2.848	2.848	2.848	3.171	3.171	3.171	3.171	3.171

```r
ggplot(res, aes(x = init, y = root)) + geom_point() +
scale_x_continuous(breaks = round(seq(min(res$init), max(res$init), by = 1),1)) +
labs(x = "Initial Values", y = "Roots from Newton-Raphson") +
theme(plot.title = element_text(hjust = 0.5)) +
ggtitle("Scatter Plot of Roots vs. Initial Values")
```

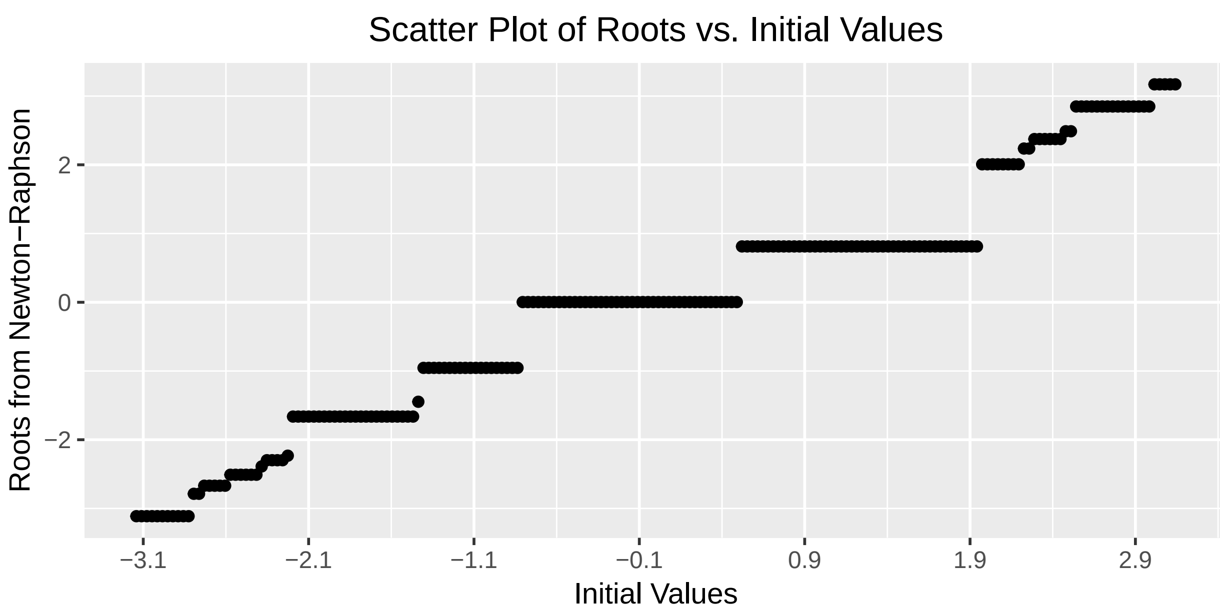

Figure 2: Scatter Plot of Roots vs. Initial Values

```r
table <- as.data.frame(table(res[,2]))
table <- t(as.matrix(table))
table <- rbind(table, cumsum(table[2,]))
rownames(table) <- c("Roots", "Amount", "Cumulative Amount")
pander(table, split.table = 120, style = 'rmarkdown', caption = "Roots, Amount and Cumulative Amount")
```

Table 25: Roots, Amount and Cumulative Amount (continued below)

Roots	-3.11247	-2.78656	-2.66886	-2.50936	-2.38826	-2.29793	-2.29792	-2.23219
Amount	11	2	5	6	1	3	1	1
Cumulative Amount	11	13	18	24	25	28	29	30

Table 26: Table continues below

Roots	-1.66271	-1.4475	-0.95441	-0.9544	0.00312	0.81264	2.00722	2.237
Amount	24	1	14	5	42	46	8	1
Cumulative Amount	54	55	69	74	116	162	170	171

Roots	2.23701	2.37471	2.48845	2.84842	3.17071
Amount	1	6	2	15	5
Cumulative Amount	172	178	180	195	200

From the table above, we can partition the 200 initial points into 21 separate groups with each group corresponding to a separate unique outcome of the optimization. According to order number of initial values, the 21 groups are

$[1, 11]$	$[12, 13]$	$[14, 18]$	$[19, 24]$	$[25, 25]$	$[26, 28]$
$[29, 29]$	$[30, 30]$	$[31, 54]$	$[55, 55]$	$[56, 69]$	$[70, 74]$
$[75, 116]$	$[117, 162]$	$[163, 170]$	$[171, 171]$	$[172, 172]$	$[173, 178]$
$[179, 180]$	$[181, 195]$	$[196, 200]$			

Table 28: Groups of Initial Values for the Same Root

2 Modeling Beetle Data

2.1 Least Square Method

The solution for the differential equation is not unique unless we give a initial condition which will determine N_0. In the following solutions, we assume $N_0 = 2$. This is reasonable since at time 0 the population size is 2. We need to minimize

$$g(r, K) = \sum_{i=1}^{n} \left[N_i - \frac{2K}{2 + (K-2)e^{-rt_i}} \right]^2$$

We use Gauss-Newton approach to address this optimization problem, this is also the default method of the function nlxb in package nlmrt. The initial values for both K and r are 1. The roots and contor plot are below

```
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))

goal <- function(r, K){
  goal <- 0
  for (i in 1:nrow(beetles)) {
    goal <- goal + (beetles[i,2]-(2 * K)/(2 + (K - 2) * exp(-r * beetles[i,1])))^2
  }
  goal
}

library("ggplot2")
r <- seq(1, 2000, 0.1)
K <- seq(1, 2000, 0.1)
data <- data.frame(r = r, K = K)
data$z <- with(data, goal(r, K))
ggplot(data, aes(r, K, z))+ geom_point(aes(colour = z)) + stat_density2d() +
theme(plot.title = element_text(hjust = 0.5)) +
ggtitle("Contour Plot for the Objective Function")
```

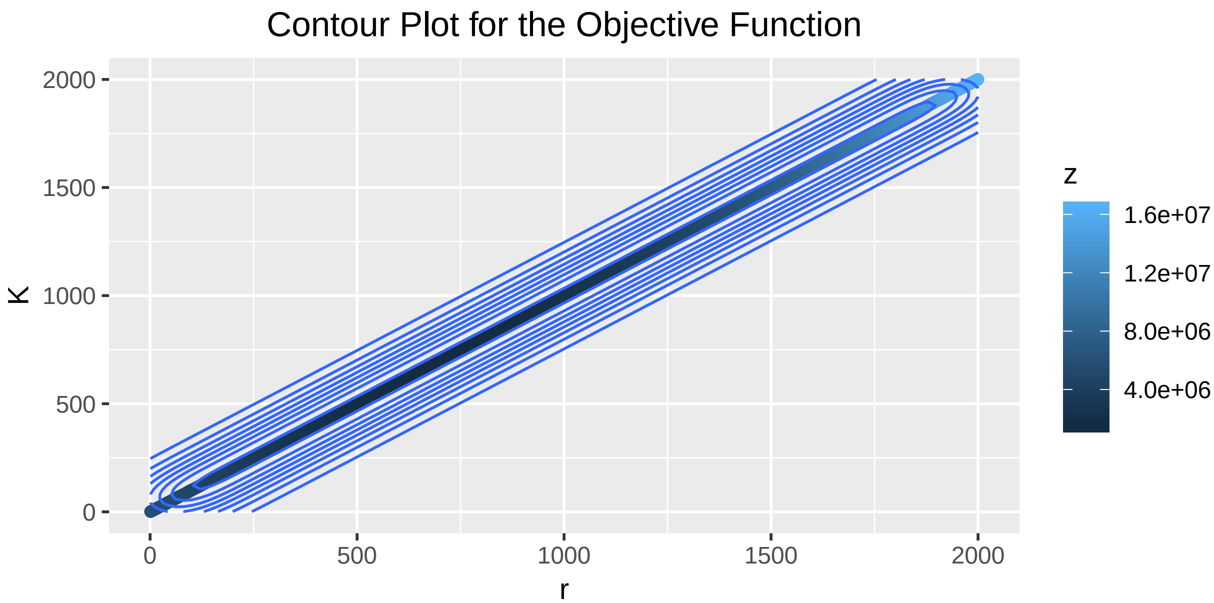

Figure 3: Contour Plot for the Objective Function

```
library("nlmrt")
mod.data <- data.frame(x = beetles[,1], y = beetles[,2])
nlmod <- nlxb(y ~ 2 * K/(2 + (K - 2) * exp(-r * x)), data = mod.data, start = c(K = 1, r = 1))
nlmod$coefficients
```

```
##        K        r
##  709.222 8748.839
```

2.2 MLE Method

Since we assume $\log N_t \sim N(\log f(t), \sigma^2)$, the likelihood function is

$$L(r, K, \sigma^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(\log N_i - \log \frac{2K}{2+(K-2)\exp(-rt_i)})^2}{2\sigma^2})$$

Thus, the log likelihood function is

$$l(r, K, \sigma^2) = -\frac{n}{2}\log(2\pi\sigma^2) - \sum_{i=1}^{n} \frac{(\log N_i - \log \frac{2K}{2+(K-2)\exp(-rt_i)})^2}{2\sigma^2}$$

We would like to maximize the log likelihood function regarding to r, K, σ^2. Since the objective function is very complex, directly using Newton-Raphson method is not appropriate. We instead use Quasi-Newton method, specifically, we use BFGS method in package optimx or optim. The point estimations and variance for it are shown below

```
library("optimx")
goal <- function(x){
  goal <- 0
  r <- x[1]
  K <- x[2]
  sigma2 <- x[3]
  for (i in 1:nrow(beetles)) {
```

```r
    goal <- goal - log(2 * pi * sigma2)/2 -
      (log(beetles[i,2]) - log((2 * K)/(2 + (K - 2) * exp(-r * beetles[i,1]))))^2/(2 * sigma2)
  }
  goal
}

BFGSmod <- optimx(c(800, 800, 70), fn = goal, gr = NULL, method = "BFGS", hessian = TRUE)
coef.BFGSmod <- coef(BFGSmod)
colnames(coef.BFGSmod) <- c("r", "K", "sigma^2")
BFGSmod
```

```
##       p1      p2      p3    value fevals gevals niter convcode kkt1  kkt2
## BFGS 800 833.359 31430.3 -60.9672     16     15    NA        0 TRUE FALSE
##      xtimes
## BFGS  0.022
```

```r
coef.BFGSmod
```

```
##        r       K sigma^2
## BFGS 800 833.359 31430.3
```

```r
BFGSmod1 <- optim(c(800, 800, 70), fn = goal, gr = NULL, method = "BFGS", hessian = TRUE)
BFGSmod1$hessian
```

```
##      [,1]         [,2]         [,3]
## [1,]    0  0.00000e+00  0.00000e+00
## [2,]    0 -3.55271e-09 -3.55271e-09
## [3,]    0 -3.55271e-09 -8.88178e-09
```

```r
MASS::ginv(-BFGSmod1$hessian)
```

```
##      [,1]       [,2]       [,3]
## [1,]    0          0          0
## [2,]    0  469124961 -187649984
## [3,]    0 -187649984  187649984
```