Many local maxima and Modeling beetle data

Yuance He

09/28/2018

3.3.2

1.

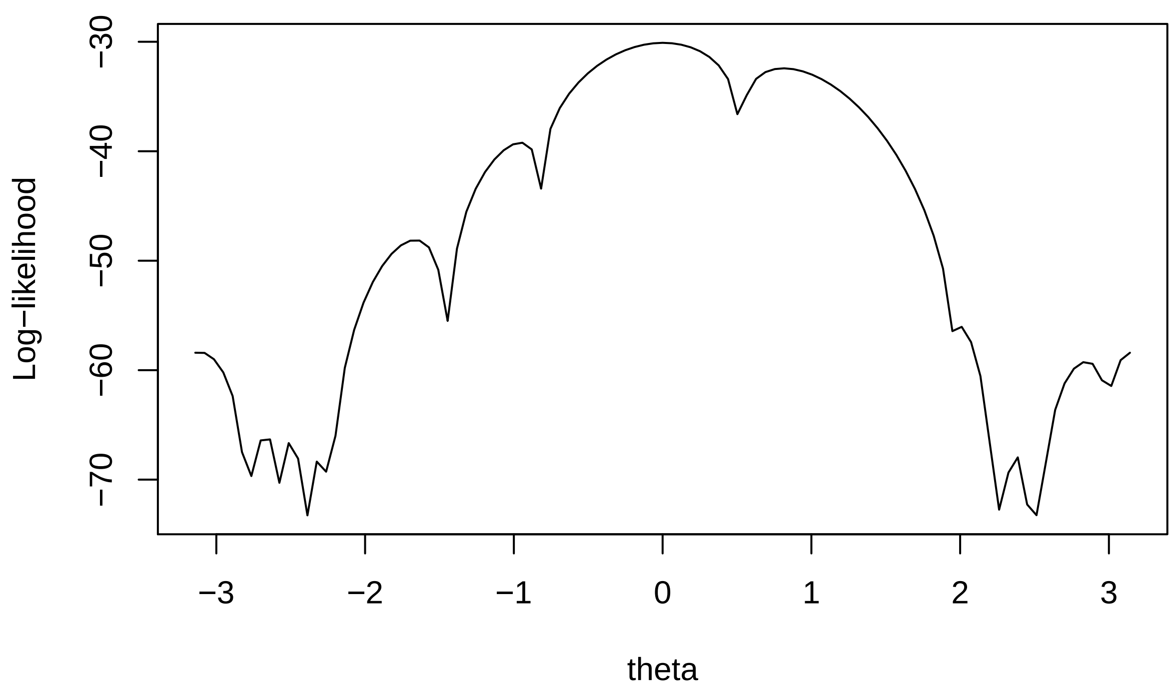
Given the density function, the likelihood function is:

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

Then the log-likelihood function is:

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

```
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)
lg <- function(theta){
  l1=-19*log(2*pi) + sum(log(1 - cos(x - theta)))
  l1
}
lf <- Vectorize(lg)
curve(lf, from = -pi, to = pi, xname = "theta", ylab = "Log-likelihood")
```



2.

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

$\bar{X}_n$ is 3.236842.

```
theta1=asin(3.236842-pi)
theta2=pi-theta1
theta1
```

```
## [1] 0.09539396
```

```
theta2
```

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

3.

```
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)
lg1 <- function(theta){
  sum( sin(x-theta)/(1-cos(x-theta)) )
}
lg2 <- function(theta){
  sum(1/(1-cos(x-theta))^2)
}

start <- c(theta1,theta2)
Newton <- function(start, max, tol = 1e-5){
  sp = start
   for(i in 1:max)
   {
      update = sp - lg1(sp)/lg2(sp)
      if(abs(update -sp) < tol) break
      sp = update
   }
  return( c(sp, i ) )
}
result = matrix(0, 2, 2)
for(i in 1:2)
{
   result[i,] = Newton(start[i], 100)
}
colnames(result) = c('Root', '# of iteration')
rownames(result) = c(theta1,theta2)

knitr::kable(result)
```

	Root	# of iteration
0.0953939615619148	0.0031499	27
3.04619869202788	3.0569835	100

4.

```
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)
lg1 <- function(theta){
  sum( sin(x-theta)/(1-cos(x-theta)) )
}
lg2 <- function(theta){
```

```r
  sum(1/(1-cos(x-theta))^2)
}

start <- c(-2.7,2.7)
Newton <- function(start, max, tol = 1e-5){
  sp = start
   for(i in 1:max)
   {
      update = sp - lg1(sp)/lg2(sp)
      if(abs(update -sp) < tol) break
      sp = update
   }
  return( c(sp, i ) )
}
result = matrix(0, 2, 2)
for(i in 1:2)
{
   result[i,] = Newton(start[i], 100)
}
colnames(result) = c('Root', '# of iteration')
rownames(result) = c(-2.7,2.7)

knitr::kable(result)
```

	Root	# of iteration
-2.7	-2.694833	100
2.7	2.839017	100

5.

```r
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)
lg1 <- function(theta){
  sum( sin(x-theta)/(1-cos(x-theta)) )
}
lg2 <- function(theta){
  sum(1/(1-cos(x-theta))^2)
}

start <- c(seq(-pi,pi, length=200))
Newton <- function(start, max, tol = 1e-5){
  sp = start
   for(i in 1:max)
   {
```

```r
      update = sp - lg1(sp)/lg2(sp)
      if(abs(update -sp) < tol) break
      sp = update
   }
  return( c(sp, i ) )
}
result = matrix(0, 200, 2)
for(i in 1:200)
{
   result[i,] = Newton(start[i], 1000)
}
colnames(result) = c('Root', '# of iteration')
rownames(result) = c(seq(-pi,pi, length=200))

knitr::kable(result)
```

	Root	# of iteration
-3.14159265358979	-3.1127852	161
-3.11001885807633	-3.1121571	65
-3.07844506256286	-3.1121563	137
-3.0468712670494	-3.1121521	151
-3.01529747153593	-3.1121534	161
-2.98372367602247	-3.1121601	172
-2.952149880509	-3.1121588	186
-2.92057608499554	-3.1121557	210
-2.88900228948207	-3.1121594	266
-2.85742849396861	-3.1121530	478
-2.82585469845514	-2.8258547	1
-2.79428090294168	-2.7942809	1
-2.76270710742821	-2.7627071	1
-2.73113331191474	-2.7311333	1
-2.69955951640128	-2.6710898	751
-2.66798572088781	-2.6679857	1
-2.63641192537435	-2.6666456	753
-2.60483812986088	-2.6048381	1
-2.57326433434742	-2.5732643	1
-2.54169053883395	-2.5130639	1000
-2.51011674332049	-2.5101167	1
-2.47854294780702	-2.5071624	540
-2.44696915229356	-2.5071666	908
-2.41539535678009	-2.4153954	1
-2.38382156126663	-2.3838216	1
-2.35224776575316	-2.3522478	1
-2.3206739702397	-2.3019260	684
-2.28910017472623	-2.2920162	257
-2.25752637921277	-2.2575264	1
-2.2259525836993	-2.2259526	1

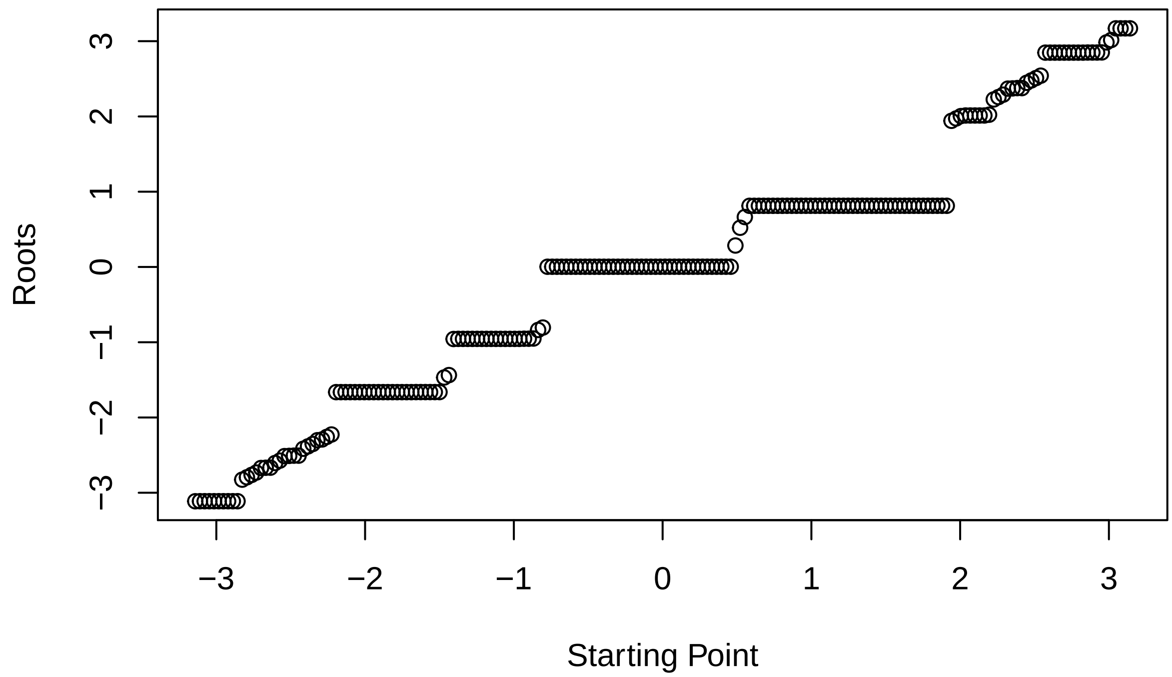
	Root	# of iteration
-2.19437878818584	-1.6630354	791
-2.16280499267237	-1.6630389	300
-2.1312311971589	-1.6630387	221
-2.09965740164544	-1.6630412	194
-2.06808360613197	-1.6630371	182
-2.03650981061851	-1.6630378	175
-2.00493601510504	-1.6630341	171
-1.97336221959158	-1.6630339	168
-1.94178842407811	-1.6630413	165
-1.91021462856465	-1.6630432	163
-1.87864083305118	-1.6630375	162
-1.84706703753772	-1.6630426	160
-1.81549324202425	-1.6630367	159
-1.78391944651079	-1.6630369	157
-1.75234565099732	-1.6630369	154
-1.72077185548386	-1.6630406	148
-1.68919805997039	-1.6630431	133
-1.65762426445693	-1.6623800	93
-1.62605046894346	-1.6623867	179
-1.59447667343	-1.6623780	230
-1.56290287791653	-1.6623807	298
-1.53132908240306	-1.6623789	430
-1.4997552868896	-1.6623831	908
-1.46818149137613	-1.4681815	1
-1.43660769586267	-1.4366077	1
-1.4050339003492	-0.9572486	1000
-1.37346010483574	-0.9553133	488
-1.34188630932227	-0.9553136	390
-1.31031251380881	-0.9553216	356
-1.27873871829534	-0.9553181	341
-1.24716492278188	-0.9553133	333
-1.21559112726841	-0.9553199	327
-1.18401733175495	-0.9553127	324
-1.15244353624148	-0.9553136	321
-1.12086974072802	-0.9553176	318
-1.08929594521455	-0.9553197	315
-1.05772214970109	-0.9553210	311
-1.02614835418762	-0.9553196	304
-0.994574558674155	-0.9553182	286
-0.96300076316069	-0.9553159	194
-0.931426967647225	-0.9534545	360
-0.89985317213376	-0.9534474	602
-0.868279376620294	-0.9510239	1000
-0.836705581106829	-0.8367056	1
-0.805131785593364	-0.8051318	1
-0.773557990079899	0.0030904	318

	Root	# of iteration
-0.741984194566434	0.0030849	124
-0.710410399052968	0.0030910	74
-0.678836603539503	0.0030898	54
-0.647262808026038	0.0030871	44
-0.615689012512572	0.0030919	39
-0.584115216999107	0.0030863	35
-0.552541421485642	0.0030895	33
-0.520967625972176	0.0030858	31
-0.489393830458711	0.0030883	30
-0.457820034945246	0.0030878	29
-0.426246239431781	0.0030854	28
-0.394672443918316	0.0030918	28
-0.363098648404851	0.0030875	27
-0.331524852891385	0.0030920	27
-0.29995105737792	0.0030865	26
-0.268377261864455	0.0030904	26
-0.236803466350989	0.0030839	25
-0.205229670837524	0.0030878	25
-0.173655875324059	0.0030914	25
-0.142082079810594	0.0030853	24
-0.110508284297128	0.0030901	24
-0.0789344887836632	0.0030863	23
-0.047360693270198	0.0030865	22
-0.0157868977567324	0.0030905	20
0.0157868977567328	0.0031501	19
0.047360693270198	0.0031445	24
0.0789344887836632	0.0031489	26
0.110508284297128	0.0031495	28
0.142082079810594	0.0031497	30
0.173655875324059	0.0031513	32
0.205229670837524	0.0031457	35
0.236803466350989	0.0031453	38
0.268377261864455	0.0031436	42
0.29995105737792	0.0031451	47
0.331524852891385	0.0031508	54
0.363098648404851	0.0031520	66
0.394672443918316	0.0031499	88
0.426246239431781	0.0031486	135
0.457820034945246	0.0031476	271
0.489393830458712	0.2850006	1000
0.520967625972177	0.5209676	1
0.552541421485642	0.6633106	1000
0.584115216999107	0.8124873	402
0.615689012512572	0.8124927	244
0.647262808026038	0.8124903	181
0.678836603539503	0.8124878	147

	Root	# of iteration
0.710410399052968	0.8124907	125
0.741984194566434	0.8124965	108
0.773557990079899	0.8124901	90
0.805131785593364	0.8124959	60
0.83670558110683	0.8127866	71
0.868279376620294	0.8127860	79
0.89985317213376	0.8127859	82
0.931426967647226	0.8127807	84
0.96300076316069	0.8127789	85
0.994574558674156	0.8127836	85
1.02614835418762	0.8127866	85
1.05772214970109	0.8127786	86
1.08929594521455	0.8127800	86
1.12086974072802	0.8127813	86
1.15244353624148	0.8127827	86
1.18401733175495	0.8127843	86
1.21559112726841	0.8127860	86
1.24716492278188	0.8127777	87
1.27873871829534	0.8127792	87
1.31031251380881	0.8127806	87
1.34188630932227	0.8127823	87
1.37346010483574	0.8127844	87
1.4050339003492	0.8127869	87
1.43660769586267	0.8127792	88
1.46818149137613	0.8127816	88
1.4997552868896	0.8127848	88
1.53132908240307	0.8127786	89
1.56290287791653	0.8127823	89
1.59447667343	0.8127778	90
1.62605046894346	0.8127835	90
1.65762426445693	0.8127813	91
1.68919805997039	0.8127815	92
1.72077185548386	0.8127860	93
1.75234565099732	0.8127868	95
1.78391944651079	0.8127795	99
1.81549324202425	0.8127777	105
1.84706703753772	0.8127873	116
1.87864083305118	0.8127821	148
1.91021462856465	0.8127798	278
1.94178842407811	1.9417884	1
1.97336221959158	1.9733622	1
2.00493601510504	2.0049360	1
2.03650981061851	2.0127552	614
2.06808360613197	2.0127549	698
2.09965740164544	2.0127588	726
2.1312311971589	2.0127616	758

	Root	# of iteration
2.16280499267237	2.0127599	842
2.19437878818584	2.0230218	1000
2.2259525836993	2.2259526	1
2.25752637921277	2.2575264	1
2.28910017472623	2.2891002	1
2.3206739702397	2.3697441	1000
2.35224776575316	2.3727307	539
2.38382156126663	2.3766770	308
2.41539535678009	2.3766735	842
2.44696915229356	2.4469692	1
2.47854294780702	2.4785429	1
2.51011674332049	2.5101167	1
2.54169053883395	2.5416905	1
2.57326433434742	2.8478597	817
2.60483812986088	2.8478624	393
2.63641192537435	2.8478622	304
2.66798572088781	2.8478612	269
2.69955951640128	2.8478577	250
2.73113331191474	2.8478621	238
2.76270710742821	2.8478572	227
2.79428090294168	2.8478630	215
2.82585469845514	2.8478614	187
2.85742849396861	2.8489887	167
2.88900228948207	2.8489877	315
2.92057608499554	2.8489822	493
2.952149880509	2.8505105	1000
2.98372367602247	2.9837237	1
3.01529747153593	3.0152975	1
3.0468712670494	3.1703999	497
3.07844506256286	3.1703956	296
3.11001885807633	3.1703957	216
3.14159265358979	3.1704001	161

```
plot(seq(-pi, pi, length = 200), result[,1], xlab = "Starting Point", ylab = "Roots")
```



Then we can divide the set of the number of starting values into; [1:10] [11] [12] [13] [14:16] [17:23] [24] [25:29] [30] [31:54] [55] [56:70] [71:110] [111:116] [117] [118:160] [161:162] [163] [164:170] [171:173] [174:178] [179:181] [182:190] [191:193] [194] [195] [196] [196:200]

3.3.3

1.

```
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))
```

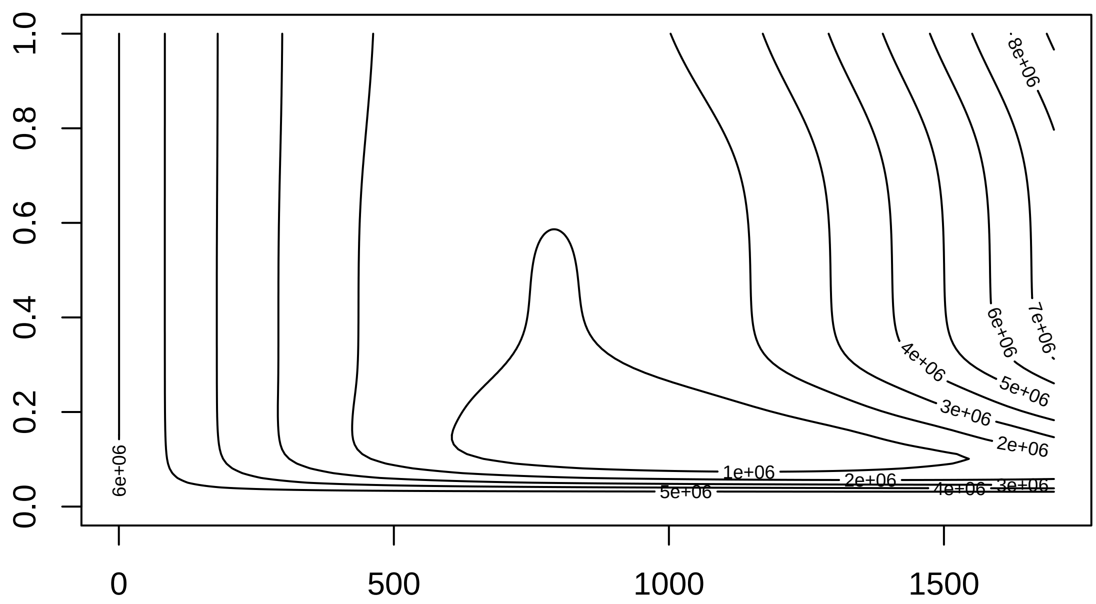
Since $N_0 = 2$, to minimize:

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

```
g <- function(k,r) {
  k*2/(2 + (k-2)*exp(-r*beetles$days))
}

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

```
  sum( (beetles$beetles - g(k, r))^2 )
}
r <- seq(0,1,length.out = 100)
k <- seq(0,1700,length.out = 10000)
m <- outer(k, r, FUN = Vectorize(ge))
contour(k, r, m)
```



2.

```
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))

lg <- function(theta) {
  K = theta[1]
  r = theta[2]
  sigma = theta[3]
  l1= -sum(dnorm(log(beetles$beetles), log((2*K)/(2 + (K - 2)*exp(-r*beetles$days))), sigma, lo
return(l1)
}
library(stats)
```

```r
options(warn=-1)
mle=optim(c(1100,0.2,3),lg,method="BFGS",hessian=TRUE)
mle
```

```
## $par
## [1] 1099.9586085    0.3919630    0.7318843
##
## $value
## [1] 11.07797
##
## $counts
## function gradient
##       53       16
##
## $convergence
## [1] 0
##
## $message
## NULL
##
## $hessian
##               [,1]         [,2]        [,3]
## [1,]  6.239453e-06  0.001680744 -0.01820238
## [2,]  1.680744e-03 86.752169945  0.04327081
## [3,] -1.820238e-02  0.043270813 37.44918186
```

Estimators:

$$\hat{k} = 1099.9586085$$

$$\hat{r} = 0.3919630$$

$$\hat{\sigma} = 0.7318843$$

Variance: var($\hat{k}$)=6.239453e-06 var($\hat{r}$)=86.752169945 var($\hat{\sigma}$)=37.44918186