MATH 534 Homework 3

Gustavo Esparza

02/17/2019

1.)
a.) Derive the following:

The log-likelihood function, $l(\theta)$:

$$f(x) = \frac{1}{2\pi}\Big[1 - cos(x-\theta)\Big]$$

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

$$l(\theta) = \sum_{i=1}^{n} log(1 - cos(x_i - \theta)) - nlog(2\pi)$$

The gradient, $l'(\theta)$

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

The Hessian, $l''(\theta)$

$$\frac{\partial^2 l}{\partial \theta^2} = \sum_{i=1}^{n} \frac{cos(x_i - \theta) - cos^2(x_i - \theta) - sin^2(x_i - \theta)}{(1 - cos(x_i - \theta))^2} = \sum_{i=1}^{n} \frac{-(1 - cos(x_i - \theta))}{(1 - cos(x_i - \theta))^2} = \sum_{i=1}^{n} \frac{-1}{1 - cos(x_i - \theta)}$$

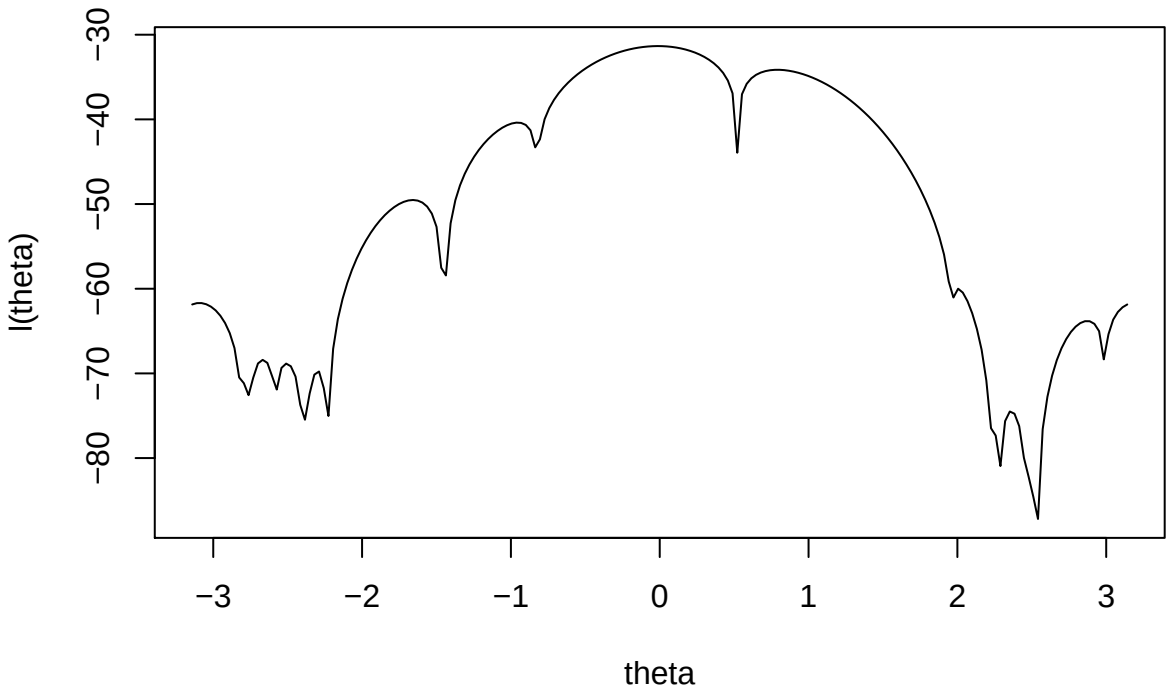
b.) Write R functions for each of the functional expressions in part **a**.

```
loglike = function (x,t) {

  sum(log((1-cos(x-t))/(2*pi)))
}


gradient = function(x,t) {
  dl= -sum(sin(x-t)/(1-cos(x-t) ) )
}


hessian=function(x,t) {
  ddl= -sum(1/(1-cos(x-t)))
}
```

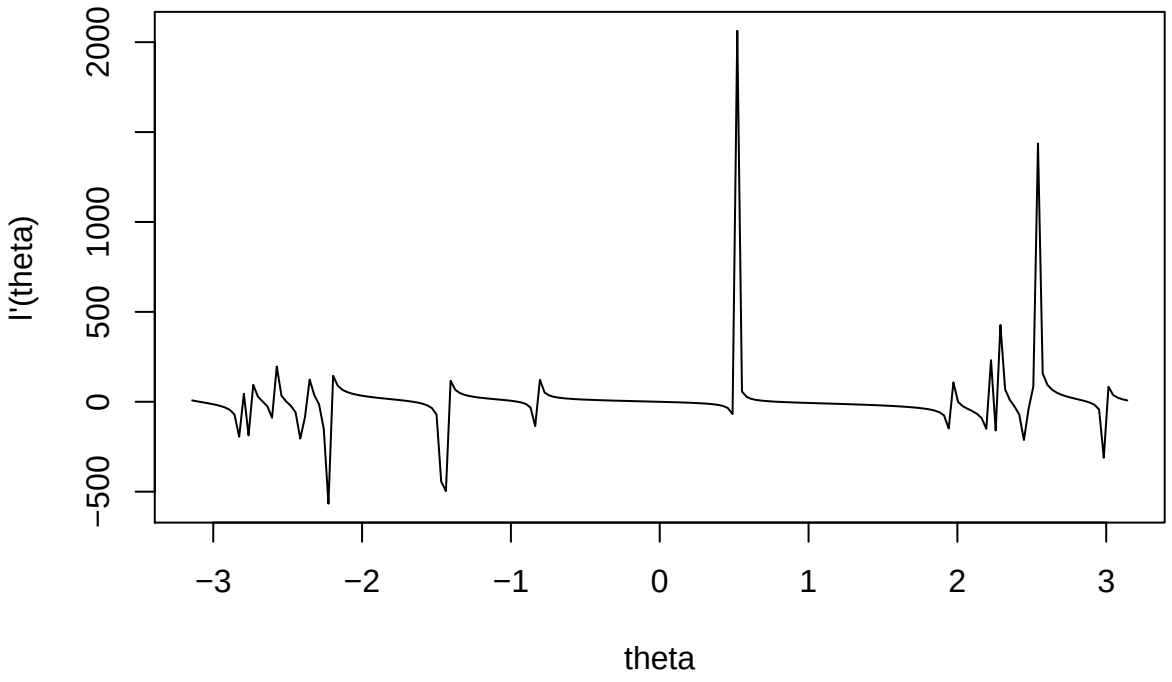
c.) Graph the log likelihood function and the gradient function.

```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, 2.50)
t=seq(-pi,pi,length=200)
```

```r
plot(t, sapply(X=t, FUN=function(t) loglike(x,t)),
     type="l",xlab = "theta",ylab="l(theta)")
```




```r
plot(t, sapply(X=t, FUN=function(t) gradient(x,t)),
     type="l",xlab="theta",ylab="l'(theta)")
```



From the graph of the log-likelihood function, we can see that there are many maximum points throughout

the given interval. The maximum of the likelihood function appears to be located between -1 and 0. Given the graph of the Gradient, we can see that there is a point where the function stopped increasing and began decreasing located near the maximum likelihood estimate shown in the first graph.This tells us that the estimate is validated by the properties of the MLE.

d.) Find the method of moments estimator of θ.

$$\mu_1 = E(x) = \int_0^{2\pi} \frac{x(1 - cos(x - \theta))}{2\pi} dx = \pi - sin(\theta)$$

$$\bar{X} = \pi - sin(\theta) \hookrightarrow \hat{\theta} = asin(\pi - \bar{X})$$

```
MOM=asin(pi-mean(x))
```

e.) Write an R function that applies the Newton method for finding the maximum of $l(\theta)$.

```
Newton <- function (maxit, x, t) {

  t_star= -0.011972
  sig = -log10( abs(t-t_star)/abs(t_star) )
  rate=0

  cat("it" , "     Theta(n)", "         CR", "  Digits\n")

  cat(sprintf('%2.0f   %12.12f    %4.2f   %2.1f\n'
              ,0,t,rate,sig))

  for (it in 1:maxit){

    dl=gradient(x,t)
    ddl=hessian(x,t)

    tnew = t - dl/ddl
    rate = abs(tnew-t_star)/abs(t-t_star)
    sig = -log10(abs(tnew-t_star)/abs(t_star))

    cat(sprintf('%2.0f   %12.12f    %4.2f   %2.1f\n',
                it,tnew,rate,sig))

    t = tnew
  }
}
```

Use the MOM estimator as the initial value and apply your function to find the MLE.

```
maxit =20
t=MOM
x = x
Newton(maxit, x, t)
```

```
## it      Theta(n)          CR   Digits
##  0   -0.058440606140    0.00   -0.6
##  1   -0.011385022018    0.01   1.3
##  2   -0.011971868104    0.00   5.0
##  3   -0.011972002287    0.02   6.7
```

```
##  4   -0.011972002287    1.00   6.7
##  5   -0.011972002287    1.00   6.7
##  6   -0.011972002287    1.00   6.7
##  7   -0.011972002287    1.00   6.7
##  8   -0.011972002287    1.00   6.7
##  9   -0.011972002287    1.00   6.7
## 10   -0.011972002287    1.00   6.7
## 11   -0.011972002287    1.00   6.7
## 12   -0.011972002287    1.00   6.7
## 13   -0.011972002287    1.00   6.7
## 14   -0.011972002287    1.00   6.7
## 15   -0.011972002287    1.00   6.7
## 16   -0.011972002287    1.00   6.7
## 17   -0.011972002287    1.00   6.7
## 18   -0.011972002287    1.00   6.7
## 19   -0.011972002287    1.00   6.7
## 20   -0.011972002287    1.00   6.7
```

Using the M.O.M. estimator as our starting value, the Newton approximation converges to a value that is very close to the Maximum of the log-likelihood function provided. This makes sense since the MOM estimator is already very close to the MLE to begin with.

What solutions do you find when you start at -2.7?

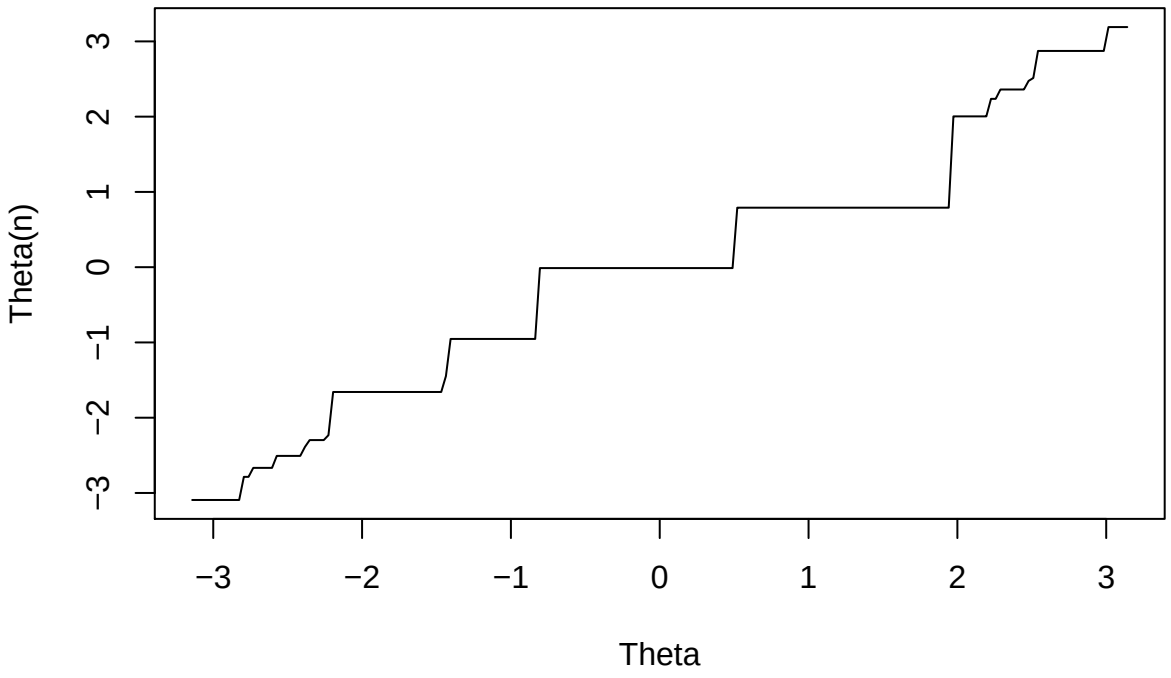
```
t=-2.7
Newton(maxit, x, t)
```

```
## it      Theta(n)          CR   Digits
##  0   -2.700000000000    0.00   -2.4
##  1   -2.674113655831    0.99   -2.3
##  2   -2.666793927068    1.00   -2.3
##  3   -2.666699927130    1.00   -2.3
##  4   -2.666699926101    1.00   -2.3
##  5   -2.666699926101    1.00   -2.3
##  6   -2.666699926101    1.00   -2.3
##  7   -2.666699926101    1.00   -2.3
##  8   -2.666699926101    1.00   -2.3
##  9   -2.666699926101    1.00   -2.3
## 10   -2.666699926101    1.00   -2.3
## 11   -2.666699926101    1.00   -2.3
## 12   -2.666699926101    1.00   -2.3
## 13   -2.666699926101    1.00   -2.3
## 14   -2.666699926101    1.00   -2.3
## 15   -2.666699926101    1.00   -2.3
## 16   -2.666699926101    1.00   -2.3
## 17   -2.666699926101    1.00   -2.3
## 18   -2.666699926101    1.00   -2.3
## 19   -2.666699926101    1.00   -2.3
## 20   -2.666699926101    1.00   -2.3
```

Using -2.7, we converge to a value that is very different from the previous solution. This is likely due to the fact that the newton function converged to the nearest maximum value rather than the global maximum.

f.) Repeat part e using 200 equally spaced starting values between $-\pi$ and π. Discuss your results.

```r
#Altering the Newton function to return the converged theta value

Newton2 = function (t) {

  maxit=20
  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, 2.50)
  for (it in 1:maxit){

    dl=gradient(x,t)
    ddl=hessian(x,t)

    tnew = t - dl/ddl
    t = tnew
  }
  return(tnew)
}

# Creating two sequences of length 200
# One contains the starting values for theta.
# The other will contain the converged theta value for each starting value.
t=seq(-pi,pi,length=200)
modeval=rep(0,200)

for(i in 1:200){
modeval[i]=Newton2(t[i])
}

#Plotting the starting values and their corresponding converged theta value.

plot(t,modeval,type='l',xlab="Theta", ylab="Theta(n)")
```



By partioning the 200 starting values into clusters of convergence, we can see that there are many different outcomes for the newton approximation depending on the initial starting value. This is examplified in the figure produced by showing different horizontal lines that show a constant convergence value for a given interval. By comparing the figure to the original graph of the Likelihood function, it is evident that each local maximum will have a specific interval of values which will converge to it. This shows how inaccurate the Newton method can be for a function that has multiple local maximum values.

2

a.) Write an R function that applies the secant method for this problem. Write the gradient as a seperate R function.

```
gradient = function(x,t) {
  dl= x[1]/(2+t) -(x[2]+x[3])/(1-t) + x[4]/t
}
```

```
secant = function(maxit,x,t0,t1,tolgrad,tolerr){

  MRE = abs(t1-t0)/max(1,abs(t1))


  cat("Iteration", "    Theta(n)", "              MRE", "        Gradient of theta(n)\n")
  cat(sprintf('%2.0f          %12.12f          %1.1e            %1.1e\n',
              0,t0,MRE,gradient(x,t0)))


  for(it in 1:maxit){
    dl0 = gradient(x,t0)
    dl1 = gradient(x,t1)

    if((abs(dl1) < tolgrad) & (MRE < tolerr)) {
      break
    }

    MRE = abs(t1-t0)/max(1,abs(t1))


    cat(sprintf('%2.0f          %12.12f          %1.1e            %1.1e\n'
                , it,t1,MRE,gradient(x,t1)))

    t_new= t1 - dl1*(t1-t0)/(dl1-dl0)
    t0 = t1
    t1 = t_new
  }

}
```

b.) Run your program using the starting values $\theta^{(0)} = 0.02$ and $\theta^{(1)} = 0.01$ and use $tolerr = 1e$ -6 and $tolgrad = 1e$ -9.

```
maxit=20
x=c(1997,907,904,32)
t0=.02
t1=.01
tolerr=1e-6
tolgrad=1e-9
secant(maxit,x,t0,t1,tolgrad,tolerr)
```

```
## Iteration     Theta(n)              MRE         Gradient of theta(n)
##  0          0.020000000000          1.0e-02            7.4e+02
##  1          0.010000000000          1.0e-02            2.4e+03
##  2          0.024561848011          1.5e-02            4.3e+02
##  3          0.027823212918          3.3e-03            2.7e+02
##  4          0.033351047002          5.5e-03            6.8e+01
```

```
##  5          0.035197558267          1.8e-03            1.3e+01
##  6          0.035646185180          4.5e-04            7.9e-01
##  7          0.035674309037          2.8e-05            9.6e-03
##  8          0.035674655571          3.5e-07            6.9e-06
```

Using the secant method, we can see that the approximation is reached in a fairly short amount of iterations. We can also see that the gradient reaches a point of crossing from increasing to decreasing within the last iterations.

c.) Re-run your function using the same starting values and convergence criteria, with different output.

```
secantc = function(maxit,x,t0,t1,tolgrad,tolerr){

  t_star = -1657/7680+sqrt(3728689)/7680
  sig = -log10(abs(t0-t_star)/abs(t_star))
  rate=0

  cat("Iteration", "    Theta(n)", "             Ratio", "        Digits\n")
  cat(sprintf('%2.0f          %12.12f          %4.2f            %2.1f\n'
              , 0,t0,rate,sig))


  for(it in 1:maxit){
    MRE = abs(t1-t0)/max(1,abs(t1))
    rate = abs(t1-t_star)/abs(t0-t_star)
    sig = -log10(abs(t1-t_star)/abs(t_star))

    dl0 = gradient(x,t0)
    dl1 = gradient(x,t1)

    if((abs(dl1) < tolgrad) & (MRE < tolerr)) {
      break
    }

    MRE = abs(t1-t0)/max(1,abs(t1))
    rate = abs(t1-t_star)/abs(t0-t_star)
    sig = -log10(abs(t1-t_star)/abs(t_star))


    cat(sprintf('%2.0f          %12.12f          %4.2f            %2.1f\n'
                , it,t1,rate,sig))

    t_new= t1 - dl1*(t1-t0)/(dl1-dl0)
    t0 = t1
    t1 = t_new
  }

}
```

```
maxit=20
x=c(1997,907,904,32)
t0=.02
t1=.01
tolerr=1e-6
tolgrad=1e-9
```

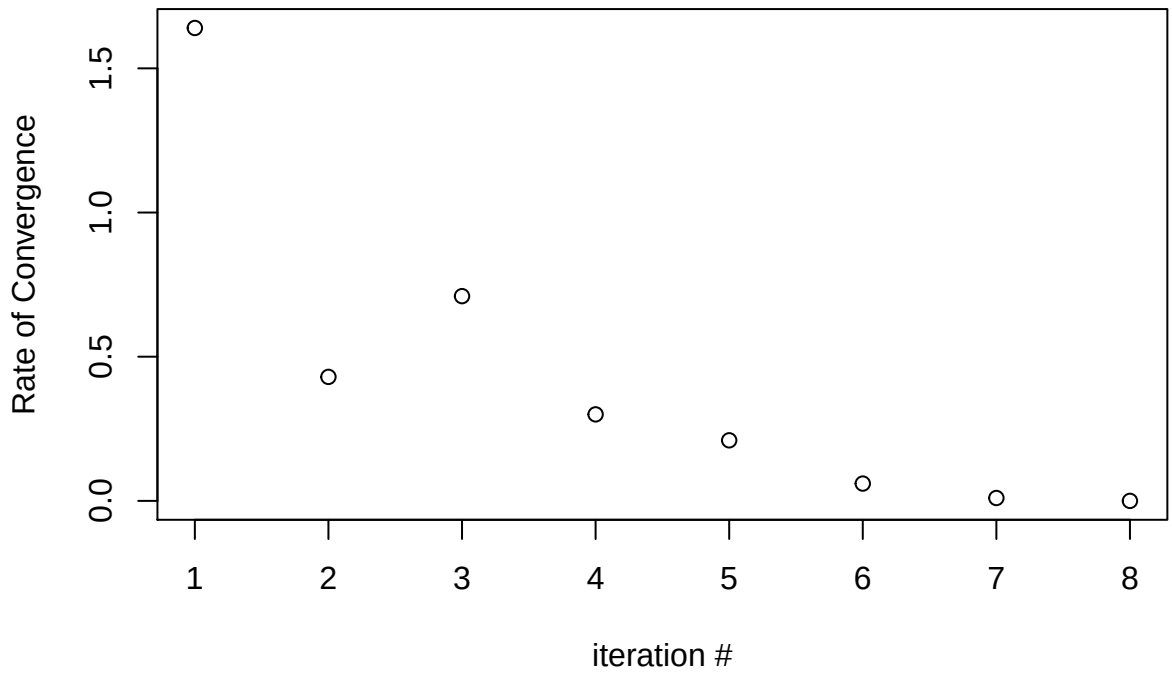
```
secantc(maxit,x,t0,t1,tolgrad,tolerr)
```

```
## Iteration     Theta(n)              Ratio         Digits
##  0          0.020000000000          0.00           0.4
##  1          0.010000000000          1.64           0.1
##  2          0.024561848011          0.43           0.5
##  3          0.027823212918          0.71           0.7
##  4          0.033351047002          0.30           1.2
##  5          0.035197558267          0.21           1.9
##  6          0.035646185180          0.06           3.1
##  7          0.035674309037          0.01           5.0
##  8          0.035674655571          0.00           8.2
```

Now observing the convergence ratio, we can see the difference between our MLE and our approximation virtually reaches zero.

d.) Based on the ratio of convergence, determine whether the secant method is linearly,quadratically. or superlinearly convergent.

```
x=seq(1,8,1)
rateval=c(1.64,.43,.71,.30,.21,.06,.01,0 )
plot(x,rateval
     ,xlab="iteration #",ylab="Rate of Convergence")
```



Again, since the convergence ratio reaches zero we can see that the secant method is superlinearly convergent. Moreso, plotting the values of the convergence ratio against the iteration value we can visually note the convergence ratio approaching zero.

e.) Repeat part (c), but use initial values, $\theta^{(0)} = .5$ and $\theta^{(1)} = .01$. Does the secant method converge for these initial values.

```
maxit=20
x=c(1997,907,904,32)
```

```
t_0=.5
t_1=.01
tolerr=1e-6
tolgrad=1e-9
secantc(maxit,x,t_0,t_1,tolgrad,tolerr)
```

```
## Iteration     Theta(n)              Ratio         Digits
##  0          0.500000000000          0.00           -1.1
##  1          0.010000000000          0.06           0.1
##  2          0.236113205980          7.81           -0.7
##  3          0.154232612419          0.59           -0.5
##  4          -0.091534935810          1.07           -0.6
##  5          -5.415293039221          42.85           -2.2
##  6          -57.393356615762          10.54           -3.2
##  7          -61.758036194809          1.08           -3.2
##  8          -118.507564138350          1.92           -3.5
##  9          -179.641949878101          1.52           -3.7
## 10          -297.549730154520          1.66           -3.9
## 11          -476.603259666372          1.60           -4.1
## 12          -773.572321173815          1.62           -4.3
## 13          -1249.599529131543          1.62           -4.5
## 14          -2022.598694332578          1.62           -4.8
## 15          -3271.626847784568          1.62           -5.0
## 16          -5293.655269176950          1.62           -5.2
## 17          -8564.712525064278          1.62           -5.4
## 18          -13857.798623401799          1.62           -5.6
## 19          -22421.942237832507          1.62           -5.8
## 20          -36279.172111427077          1.62           -6.0
```

Using these new starting values, we can clearly see that the Secant Method does not converge to any value and continues to increase.