< STAT-5361 > HW#3-Exercises

Hee-Seung, Kim

September 21, 2018

Contents

(a) Proof

$$f(x;\theta) = \frac{1}{\pi[1+(x-\theta)^2]}, x \in R, \theta \in R.$$

Because $x_1, ..., x_n$ is an i.i.d. sample and $l(\theta)$ the log-likelihood function of θ based on the sample. Therefore,

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

And,

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

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

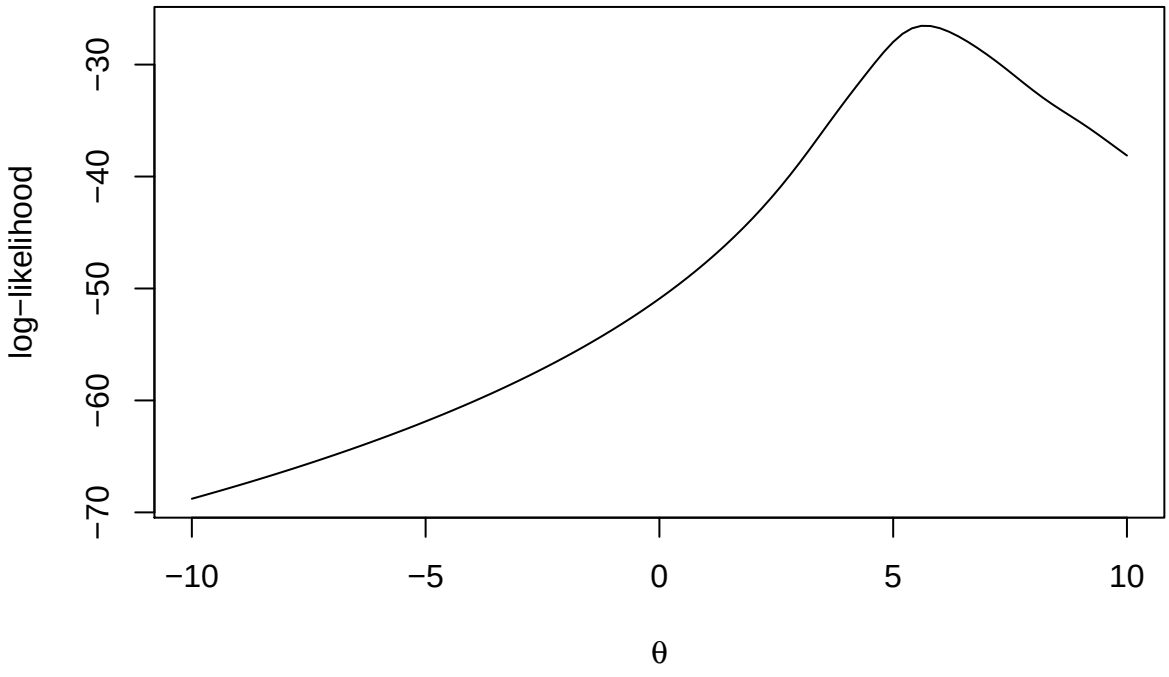
As for $I(\theta)$,

$$
I_n(\theta) = n\int \frac{f'(x)^2}{f(x)}dx
$$

$$
\begin{aligned}
I(\theta) &= n\int \frac{(f'(x))^2}{f(x)}dx \\
&= n\int_{-\infty}^{\infty} \frac{-(\frac{2\pi x}{(\pi(1+x^2))^2})^2}{\frac{1}{\pi(1+x^2)}}dx = n\int_{-\infty}^{\infty}\frac{4\pi^2x^2}{(\pi(1+x^2))^3}dx \\
&= \frac{4n}{\pi}\int_{-\infty}^{\infty}\frac{x^2}{(1+x^2)^3}dx = \frac{4n}{\pi}\int_{-\infty}^{\infty}\frac{x^2}{1+x^2}\frac{1}{(1+x^2)^2}dx \\
&= \frac{4n}{\pi}\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}}\frac{\arctan^2(\theta)}{1+\arctan^2(\theta)}\frac{1+\arctan^2(\theta)}{(1+\arctan^2(\theta))^2}d\theta \\
&= \frac{4n}{\pi}\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}}\sin^2(\theta)\cos^2(\theta)d\theta = \frac{4n}{\pi}\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}}\frac{sin^2(2\theta)}{4}d\theta \\
&= \frac{n}{\pi}(\frac{\pi}{2}) = \frac{n}{2}
\end{aligned}
$$

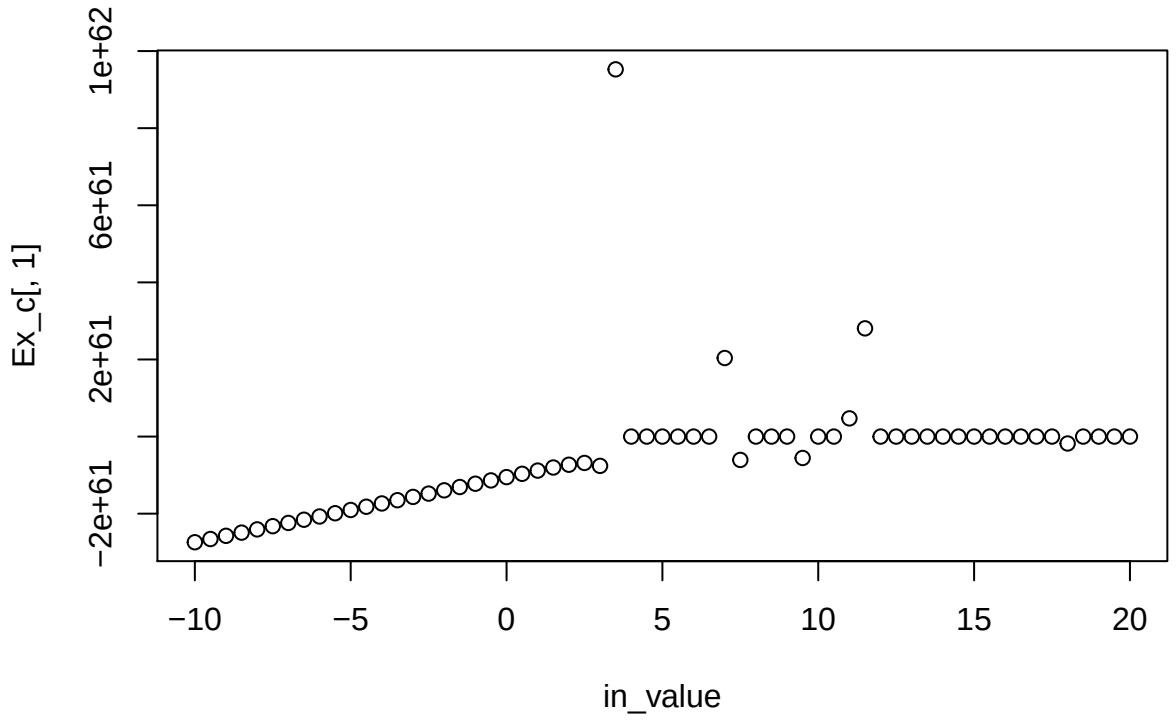
(b) Graph the log-likelihood function

```r
set.seed(20180909)
n = 10


data <- rcauchy(n,5,1)
in_value = seq(-10, 20, by=0.5);

m <- length(in_value)
y <- c()

Log_like=function(theta)
{
  Hee= -(length(data)*log(pi) + sum(log(1+(theta-data)^2))) #-log
  return(Hee)
}

curve(sapply(x,Log_like),-10,10,xlab=expression(theta), ylab='log-likelihood')
```



(C)

```
set.seed(20180909)
n = 10


data <- rcauchy(n,5,1)
in_value = seq(-10, 20, by=0.5);

m <- length(in_value)

first_deri=function(theta){
  first_deri=-2*sum((theta-data)/(1+(theta-data)^2))
  return(first_deri)
}
second_deri=function(theta){
  second_deri=-2*sum((1-(theta-data)^2)/(1+(theta-data)^2)^2)
  return(second_deri)
}

Newton = function(in_value,max, tol=1e-10){
  curr=in_value
  for(i in 1:max)
  {
    update=curr-first_deri(curr)/second_deri(curr)
    if(abs(update-curr)<tol) break
    curr=update

  }
  return(c(curr,i))
}


iii=length(in_value)
Ex_c=matrix(0,iii,2)
for(i in 1:iii){
 Ex_c[i,]=Newton(in_value[i], 200)

}

  plot(in_value,Ex_c[,1])
```




```
  Ex_c[,1]
```

```
##  [1] -2.741130e+61 -2.657995e+61 -2.574756e+61 -2.491407e+61 -2.407943e+61
##  [6] -2.324355e+61 -2.240636e+61 -2.156778e+61 -2.072774e+61 -1.988614e+61
## [11] -1.904288e+61 -1.819787e+61 -1.735103e+61 -1.650227e+61 -1.565152e+61
## [16] -1.479876e+61 -1.394404e+61 -1.308754e+61 -1.222971e+61 -1.137146e+61
## [21] -1.051467e+61 -9.663188e+60 -8.825356e+60 -8.020672e+60 -7.301183e+60
## [26] -6.841241e+60 -7.604318e+60  9.526123e+61  2.056366e+01  2.108229e+01
## [31]  5.685422e+00  5.685422e+00  5.685422e+00  5.685422e+00  2.038366e+61
## [36] -6.058665e+60  1.937744e+01  2.108229e+01  5.685422e+00 -5.551985e+60
## [41]  2.108229e+01  5.685422e+00  4.715381e+60  2.807261e+61  2.056366e+01
## [46]  2.056366e+01  2.056366e+01  2.108229e+01  2.108229e+01  2.108229e+01
## [51]  1.937744e+01  2.056366e+01  2.056366e+01  1.937744e+01  1.937744e+01
## [56]  1.937744e+01 -1.790860e+60  2.056366e+01  2.056366e+01  2.056366e+01
## [61]  2.056366e+01
```

(d)

```
 set.seed(20180909)
 n = 10


 data <- rcauchy(n,5,1)
 in_value = seq(-10, 20, by=0.5);

 m <- length(in_value)

 first_deri=function(theta){
   first_deri=-2*sum((theta-data)/(1+(theta-data)^2))
   return(first_deri)
 }

fixed_point=function(in_value,alpha, max=1000, tol=1e-10 ){

  curr=in_value
  for(i in 1: max)
  {
    update=curr + (alpha*first_deri(curr))
    if (abs(update-curr)<tol) break
    curr=update
  }
return(c(curr,i))

 }

alpha_1 <- alpha_0.64 <- alpha_0.25 <- matrix(0,length(in_value),2)

for(i in 1:length(in_value))
{
  alpha_1[i,]=fixed_point(in_value[i],1,1000)
  alpha_0.64[i,]=fixed_point(in_value[i],0.64,1000)
  alpha_0.25[i,]=fixed_point(in_value[i],0.25,1000)
}

alpha_1[,1]
```

```
##  [1] 4.087057 9.547862 4.087057 6.486114 9.547862 4.087057 6.486114
##  [8] 6.486114 9.547862 4.087057 4.087057 6.486114 9.547862 9.547862
## [15] 6.486114 4.087057 6.486114 6.486114 4.087057 9.547862 4.087057
## [22] 4.087057 4.087057 9.547862 6.486114 6.486114 6.486114 9.547862
## [29] 9.547862 9.547862 9.547862 6.486114 4.087057 4.087057 4.087057
## [36] 4.087057 4.087057 4.087057 6.486114 6.486114 6.486114 6.486114
## [43] 6.486114 6.486114 4.087057 9.547862 9.547862 9.547862 9.547862
```

```
## [50] 4.087057 4.087057 4.087057 6.486114 6.486114 6.486114 9.547862
## [57] 9.547862 4.087057 6.486114 6.486114 4.087057
```

```
alpha_0.64[,1]
```

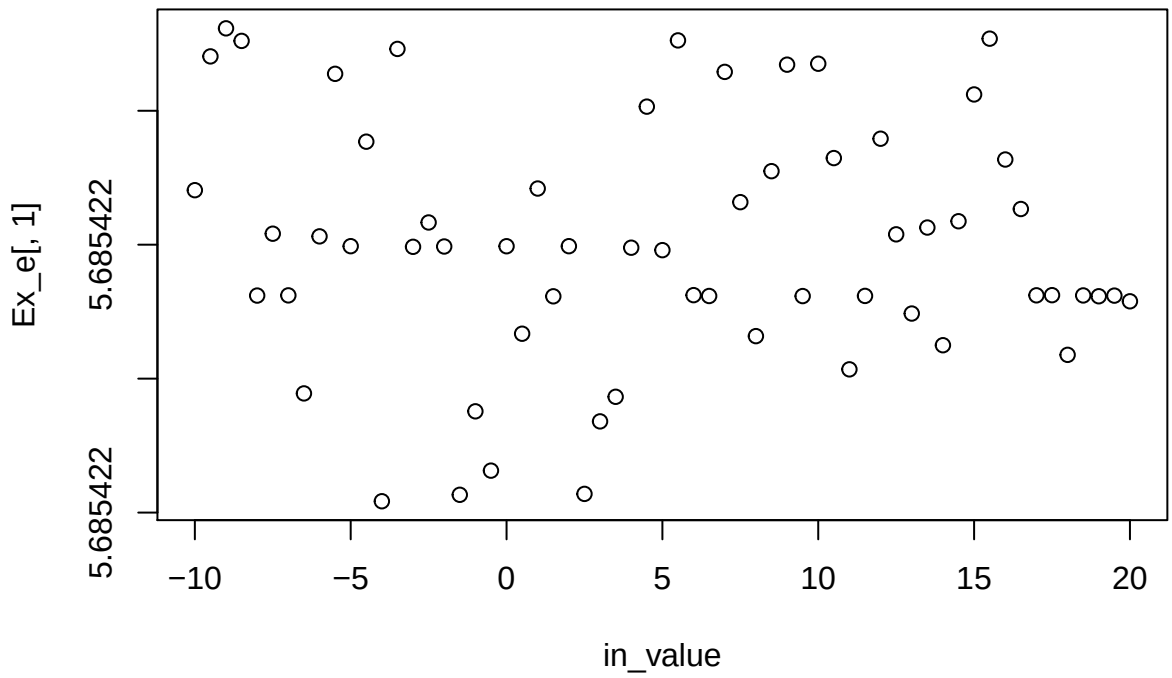
```
##  [1] 5.529107 7.706106 5.127538 4.951869 6.725152 5.197111 7.075917
##  [8] 5.964188 5.182739 4.957631 5.173272 7.716718 5.540465 5.199886
## [15] 4.951297 7.557473 7.608167 5.157557 5.181790 5.443351 7.564154
## [22] 7.183275 7.398215 6.609417 5.290579 7.154975 5.240143 5.549859
## [29] 4.951676 7.161017 4.993766 5.286890 6.371987 6.900178 7.181054
## [36] 7.203470 5.187023 5.222680 5.059510 5.045353 7.383910 7.715984
## [43] 7.607361 6.868118 7.572404 5.053074 4.975029 5.570223 6.865950
## [50] 6.966023 5.031455 5.238599 7.125650 7.319846 5.173538 5.354787
## [57] 7.156821 5.294795 7.372832 5.256387 5.266334
```

```
alpha_0.25[,1]
```

```
##  [1] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
##  [8] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [15] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [22] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [29] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [36] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [43] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [50] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [57] 5.685422 5.685422 5.685422 5.685422 5.685422
```

(e)

```
set.seed(20180909)
n = 10


data <- rcauchy(n,5,1)
in_value = seq(-10, 20, by=0.5);

m <- length(in_value)

first_deri=function(theta){
  first_deri=-2*sum((theta-data)/(1+(theta-data)^2))
  return(first_deri)
}

Newton = function(in_value,max, tol=1e-10){
  curr=in_value
  for(i in 1:max)
  {
    update=curr+2*first_deri(curr)/10
    if(abs(update-curr)<tol) break
    curr=update

  }
  return(c(curr,i))
}


iii=length(in_value)
Ex_e=matrix(0,iii,2)
for(i in 1:iii){
  Ex_e[i,]=Newton(in_value[i], 200)

}

plot(in_value,Ex_e[,1])
```




```
Ex_e[,1]
```

```
##  [1] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
##  [8] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [15] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [22] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [29] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [36] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [43] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [50] 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422 5.685422
## [57] 5.685422 5.685422 5.685422 5.685422 5.685422
```

(f) We have performed 3 iterations of the Fisher scoring method, then the Newton-Raphson method for refinement. Under Fisher scoring together with Newton method, the convergence points of MLE are less affected by choice of initial point. The number of iterations is not significantly different from the results of Newton-Raphson method alone. All iterations are similar in terms of the speed. Under those testing conditions, however, we can say that the Fisher Scoring with Newton method is the most stable algorithm for global maximization overall