Homework 5 (STAT 5860)

Your Name Here

Due by 11:59 pm, Apr. 11, 2018

Insturctions:

1. Download the `Homework5.Rmd` file from the course Elearning.
2. Open `Homework5.Rmd` in RStudio.
3. Replace the *"Your Name Here"* text in the `author` with your own name.
4. Write your answer to each problem by editing `Homework5.Rmd`.
5. After you finish all the problems, click `Knit to PDF` to create a pdf file. Upload your pdf file to Homework 5 Dropbox in the course Elearning.

Set the seed number

```r
set.seed(411)
```

Problem 1. The Cauchy distribution with scale 1 has following density function

$$f(x) = \frac{1}{\pi\left[1+(x-\eta)^2\right]}, \quad -\infty < x < \infty.$$

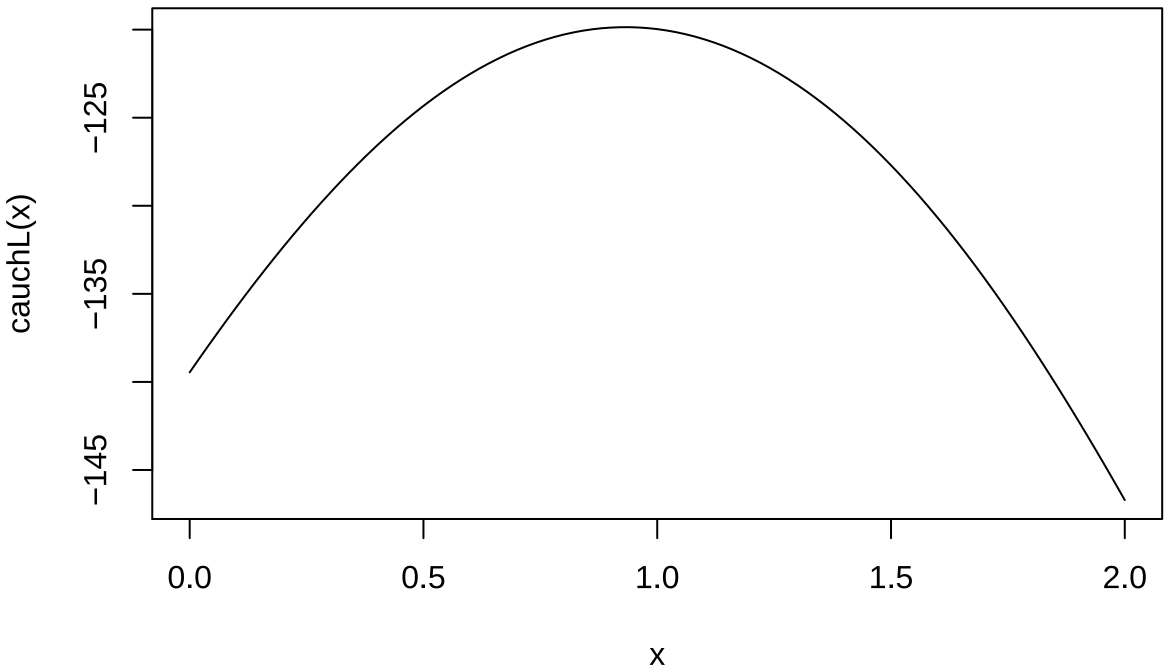
(a) Generate 100 random samples from a Cauchy distribution with $\eta = 1$. Here, η is the location parameter.

```r
samps <- rcauchy(100, location = 1)
```

(b) Treat the random samples you get from (a) as sample observations from a Cauchy distribution with an unknown η. Plot the log-likelihood function of η.

```r
cauchL <- function(etta){
  -sum(log(1+(samps - etta)^2))
}

cauchL <- Vectorize(cauchL)
curve(cauchL, xlim = c(0,2))
```



(c) Use the bisection method to find the maximum likelihood estimator of η.

```
cprime <- function(etta){
  sum((2*(samps-etta))/(1+(samps-etta)^2))
}
cprime <- Vectorize(cprime)

uniroot(cprime, c(0,4))$root
```

```
## [1] 0.9319376
```

(d) Use the Newton's method to find the maximum likelihood estimator of η.

```
cprime2 <- function(etta){
  sum((4*(samps-etta)^2 - 2*(1 + (samps - etta)^2))/((1 + (samps - etta)^2)^2))
}
cprime2 <- Vectorize(cprime2)

x <- 0

for(i in 1:10){
    xnew <- x - cprime(x)/cprime2(x)
    x <- xnew
    print(xnew, digits = 10)
}
```

```
## [1] 1.485640854
## [1] 0.9169469551
## [1] 0.9319416631
## [1] 0.9319387725
## [1] 0.9319387725
## [1] 0.9319387725
## [1] 0.9319387725
## [1] 0.9319387725
## [1] 0.9319387725
```

```
## [1] 0.9319387725
```

Problem 2. In statistics, Poisson regression is a generalized linear model form of regression analysis used to model count data. In Poisson regression the dependent variable Y is an observed count that follows the Poisson distribution. The rate λ is determined by a set of predictors $\boldsymbol{X}$. Here, we focus on simple Poisson regression model

$$\log \lambda = \beta_0 + \beta_1 X.$$

Solving for λ, we have

$$\lambda = e^{\beta_0 + \beta_1 X}.$$

For each data point, we have predictor x_i and an observed count y_i. Then the likelihood function is

$$L(\beta_0, \beta_1) = \prod_{i=1}^{n} \frac{e^{-e^{\beta_0+\beta_1 x_i}} (e^{\beta_0+\beta_1 x_i})^{y_i}}{y_i!}.$$

Our goal for the problem is obtaining the parameter estimates of Poisson regression.

(a) Load the data set uploaded in course Elearning. (Hint: To import .txt file into R, use `read.table()` function and you may need `header = T` also.)

```r
pois <- read.delim("poisson.txt")
```

(b) Use the Newton's method with initial value $\beta_0 = 0$ and $\beta_1 = 0$ to find the maximum likelihood estimator of β_0 and β_1.

```r
x <- pois$x
y <- pois$y

gradient <- function(y, x, beta0, beta1){
  gradient <- rep(0, 2)
  gradient[1] <- -sum(exp(beta0 + beta1*x))+sum(y)
  gradient[2] <- -sum(exp(beta0 + beta1*x)*x)+sum(x*y)
  return(gradient)
}

# Hessian of log-likelihood
hessian <- function(y, x, beta0, beta1){
  hessian <- matrix(0, nrow = 2, ncol = 2)
  hessian[1,1] <- -sum(exp(beta0 + beta1*x))
  hessian[2,2] <- -sum(exp(beta0 + beta1*x)*x^2)
  hessian[1,2] <- -sum(exp(beta0 + beta1*x)*x)
  hessian[2,1] <- hessian[1,2]
  return(hessian)
}

# set initial value
beta <- c(0,0)
```

```r
# run Netwon's method
for(i in 1:10){
  beta_new <- beta - solve(hessian(y = y, x = x, beta0 = beta[1], beta1 = beta[2])) %*% gradient(y = y,
  beta <- beta_new
  print(beta_new, digits = 7)
}
```

```
##            [,1]
## [1,] -0.9927826
## [2,]  0.3069818
##            [,1]
## [1,] -1.5667956
## [2,]  0.2886303
##            [,1]
## [1,] -1.5912315
## [2,]  0.2468419
##            [,1]
## [1,] -0.8671073
## [2,]  0.1746897
##             [,1]
## [1,] -0.01703712
## [2,]  0.10610804
##            [,1]
## [1,] 0.27383427
## [2,] 0.07979305
##            [,1]
## [1,] 0.30731353
## [2,] 0.07641495
##            [,1]
## [1,] 0.30786623
## [2,] 0.07635734
##            [,1]
## [1,] 0.30786640
## [2,] 0.07635732
##            [,1]
## [1,] 0.30786640
## [2,] 0.07635732
```

(c) Now try differnt initial value $\beta_0 = 0$ and $\beta_1 = 1$. Compare the result with (b).

```r
beta <- c(0,1)

for(i in 1:30){
  beta_new <- beta - solve(hessian(y = y, x = x, beta0 = beta[1], beta1 = beta[2])) %*% gradient(y = y,
  beta <- beta_new
  print(beta_new, digits = 7)
}
```

```
##            [,1]
## [1,] -0.9999994
## [2,]  1.0000000
##            [,1]
## [1,] -1.9999979
## [2,]  0.9999999
##            [,1]
```

```
## [1,] -2.9999939
## [2,]  0.9999997
##            [,1]
## [1,] -3.9999828
## [2,]  0.9999993
##           [,1]
## [1,] -4.999953
## [2,]  0.999998
##            [,1]
## [1,] -5.9998706
## [2,]  0.9999945
##            [,1]
## [1,] -6.9996476
## [2,]  0.9999851
##            [,1]
## [1,] -7.9990416
## [2,]  0.9999594
##            [,1]
## [1,] -8.9973943
## [2,]  0.9998897
##            [,1]
## [1,] -9.9929176
## [2,]  0.9997002
##             [,1]
## [1,] -10.9807565
## [2,]   0.9991854
##             [,1]
## [1,] -11.9477562
## [2,]   0.9977886
##             [,1]
## [1,] -12.8584725
## [2,]   0.9940091
##             [,1]
## [1,] -13.6188480
## [2,]   0.9838646
##             [,1]
## [1,] -13.9895377
## [2,]   0.9572156
##             [,1]
## [1,] -13.4295473
## [2,]   0.8911069
##             [,1]
## [1,] -11.0932333
## [2,]   0.7494319
##            [,1]
## [1,] -6.9868769
## [2,]  0.5313617
##            [,1]
## [1,] -3.3445485
## [2,]  0.3306311
##            [,1]
## [1,] -1.1595217
## [2,]  0.1927619
##             [,1]
```

```
## [1,] -0.08522282
## [2,]  0.11212903
##            [,1]
## [1,] 0.25898059
## [2,] 0.08124464
##            [,1]
## [1,] 0.30676878
## [2,] 0.07647152
##            [,1]
## [1,] 0.30786575
## [2,] 0.07635739
##            [,1]
## [1,] 0.30786640
## [2,] 0.07635732
##            [,1]
## [1,] 0.30786640
## [2,] 0.07635732
##            [,1]
## [1,] 0.30786640
## [2,] 0.07635732
##            [,1]
## [1,] 0.30786640
## [2,] 0.07635732
##            [,1]
## [1,] 0.30786640
## [2,] 0.07635732
##            [,1]
## [1,] 0.30786640
## [2,] 0.07635732
```

(d) When we want to run Poisson regression in R, we use `glm()` function with `family = poisson()`. Compare your parameter estimation results to `glm()` output.

```r
mod <- glm(y~x, family = poisson(), data = pois)
summary(mod)$coefficients
```

```
##               Estimate Std. Error  z value     Pr(>|z|)
## (Intercept) 0.30786640 0.28943495 1.063681 0.2874733279
## x           0.07635732 0.01730388 4.412728 0.0000102076
```

The parameter estimates using Newton's method converge to the parameter estimates using the `glm()` output, however, changing the initial values slightly resulted in having to increase the number of iterations by approximately 3 times in order to get results.