PROBLEM SET 1

Question 1

The Kolmogorov Smirnov test uses cumulative distribution statistics test the similarity of the empirical distribution of some observed data and a specified PDF, and serves as a goodness of fit test. The test statistic is created by:

$$D = \max_{i=1:n} \left\{ \frac{i}{n} - F_{(i)}, F_{(i)} - \frac{i-n}{n} \right\}$$

where F is the theoretical cumulative distribution of the distribution being tested and $F_{(i)}$ is the ith ordered value. Intuitively, the statistic takes the largest absolute difference between the two distribution functions across all x values. Large values indicate dissimilarity and the rejection of the hypothesis that the empirical distribution matches the queried theoretical distribution. The p value is calculated from the Kolmogorov Smirnoff CDF:

$$p(D \leq x) \frac{\sqrt{2\pi}}{x} \sum_{k=1}^{\infty} e^{-(2k-1)^2 \pi^2 / 8x^2}$$

which generally requires approximation methods (see Marsaglia, Tsang, and Wang 2003).
This so called non parametric test (this label comes from the fact that the distribution of the test statistic does not depend on the distribution of the data being tested) performs poorly in small samples, but works well in a simulation environment. Write an R function that implements this test where the reference distribution is normal. Using R generate 1,000 Cauchy random variables (`rcauchy(1000, location = 0, scale = 1)`) and perform the test (remember, use the same seed, something like `set.seed(123)`, whenever you're generating your own data).
Write an R function that implements this test where the reference distribution is normal. As a hint, you can create the empirical distribution and theoretical CDF using this code:

```
# create   empirical   distribution   of  observed data
ECDF <-  ecdf ( data )
empiricalCDF <-  ECDF( data )
# generate  test   st at is ti c
D <-  max(abs(empiricalCDF - pnorm( empirical ) ) )
```

Solution

Generating the data

```
set.seed(2314)
data=rcauchy(1000, location = 0, scale = 1)
head(data,100)
```

```
##   [1] -4.081773e+00 -4.777417e-01  1.281381e+00  7.470030e-01 -5.214961e+00
##   [6]  1.205527e+00  2.474216e-01  3.282559e+00  8.276652e-01  1.954214e+00
##  [11] -3.326768e+00  8.204078e-01  1.093578e+00  3.987004e-01 -1.931752e+00
##  [16] -1.369277e+00  1.411978e+00 -1.396051e+00  2.773598e+00 -1.197877e+00
##  [21]  6.269464e+00 -6.625546e-01  1.198101e-01  7.357933e-01 -1.617079e-01
```

```
##  [26] -1.297726e+00  4.298691e-01 -7.593109e-01  7.249070e-01 -4.888316e+00
##  [31] -2.424002e-01  1.918641e-02 -3.196469e+00 -1.342769e+00 -1.143779e+00
##  [36]  4.444557e-01 -1.128136e-01  1.142774e+00  7.646812e-01 -3.990208e-01
##  [41]  1.390504e+00  4.988478e-01  2.758791e-01  5.828149e-01  1.030180e+00
##  [46]  1.382744e+00  1.550585e+00 -6.337404e-01 -6.412342e-02  6.614687e-04
##  [51] -5.753447e-01  7.595799e-01 -2.624935e-01  5.277230e-01  5.205586e-01
##  [56]  1.820895e+00  1.055377e-01 -1.385029e+00 -4.271401e-02 -1.836304e+00
##  [61] -1.456290e+00  7.990635e-01  1.538762e-01 -5.865394e+00 -1.742805e+00
##  [66] -2.747368e+00  8.669048e-01 -7.238133e-01  3.455779e+00 -1.364538e+00
##  [71] -3.071212e+00 -2.743884e-03 -1.140853e+01 -1.837993e+02 -7.066863e+00
##  [76] -3.569025e+00 -1.045970e+01  4.700401e-01  9.967104e-01 -2.291490e+00
##  [81]  7.765982e+00 -8.284796e-01  1.830825e+00  2.473630e+00  1.117081e+00
##  [86]  1.110663e+00 -4.912867e-03  1.030137e+00 -1.710084e-01 -5.314905e-01
##  [91]  1.154841e+00 -5.527993e-01  2.693453e+00 -3.040602e-01 -4.082973e-01
##  [96]  7.418101e-01  6.877446e-01 -1.141021e-02 -1.238997e+00 -9.097828e-01
```

```
# create    empirical   distribution    of  observed data
ECDF <-  ecdf ( data )
empiricalCDF <-  ECDF( data )
# generate  test    st at is ti c
D <-  max(abs(empiricalCDF - pnorm( empiricalCDF ) ) )
head(D)
```

```
## [1] 0.4993989
```

```
head(empiricalCDF,100)
```

```
##   [1] 0.069 0.348 0.790 0.694 0.056 0.778 0.565 0.914 0.713 0.854 0.087 0.712
##  [13] 0.757 0.603 0.138 0.189 0.807 0.183 0.904 0.211 0.951 0.302 0.525 0.689
##  [25] 0.436 0.198 0.610 0.281 0.684 0.061 0.409 0.494 0.089 0.194 0.219 0.618
##  [37] 0.454 0.772 0.699 0.364 0.803 0.635 0.570 0.657 0.748 0.801 0.822 0.306
##  [49] 0.474 0.491 0.316 0.697 0.406 0.643 0.641 0.841 0.521 0.185 0.482 0.146
##  [61] 0.181 0.708 0.537 0.047 0.153 0.103 0.717 0.286 0.918 0.191 0.091 0.487
##  [73] 0.023 0.004 0.035 0.078 0.028 0.624 0.734 0.121 0.959 0.267 0.843 0.891
##  [85] 0.765 0.761 0.486 0.747 0.432 0.331 0.774 0.326 0.897 0.390 0.362 0.691
##  [97] 0.675 0.485 0.204 0.247
```

Kolmogrov testing

```
ks.test(data, "pweibull", shape=2, scale=1)
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  data
## D = 0.51716, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

Since the p-value is less than 0.05, it is significant. Hence we reject the hypothesis that the empirical distribution matches the queried theoretical distribution.

Question 2

Estimate an OLS regression in R that uses the Newton-Raphson algorithm (specifically BFGS, which is a quasi-Newton method), and show that you get the equivalent results to using lm. Use the code below to create your data.

```
set.seed (2314)
data2 <- data.frame(x = runif (200 , 1 , 10) )
data2$y <- 0 + 2.75*data2$x + rnorm(200 , 0 ,   1.5)
```

Solution

```
set.seed (2314)
data2 <- data.frame(x = runif (200 , 1 , 10) )
data2$y <- 0 + 2.75*data2$x + rnorm(200 , 0 ,   1.5)
head(data2, 40)
```

```
##           x         y
## 1  6.188292 17.909511
## 2  8.723212 25.003485
## 3  3.601562  8.520943
## 4  2.837992  8.431062
## 5  6.042752 15.893353
## 6  3.516193  9.575077
## 7  1.694856  4.600457
## 8  4.652857 12.723907
## 9  2.980669  7.807877
## 10 4.145025 10.792051
## 11 6.336518 17.000865
## 12 2.968286  7.525290
## 13 3.377963  8.551916
## 14 2.086859  4.050195
## 15 6.868451 15.936912
## 16 7.307056 16.888042
## 17 3.734644  9.407899
## 18 7.280713 21.917888
## 19 4.508680 13.711801
## 20 7.492774 22.641944
## 21 5.046874 13.458002
## 22 8.323668 21.586347
## 23 1.341602  4.727969
## 24 2.817269 10.290953
## 25 9.540717 26.252447
## 26 7.380854 19.008239
## 27 2.163069  5.815821
## 28 8.139510 25.355599
## 29 2.796931  6.754473
## 30 6.078072 15.737086
## 31 9.318715 26.116107
## 32 1.054958  4.929648
## 33 6.368602 17.726280
## 34 7.333808 19.526424
## 35 7.558152 21.569066
## 36 2.198151  5.071918
## 37 9.678174 26.634237
## 38 3.440601 10.335442
## 39 2.870223  6.443463
## 40 8.912349 23.964777
```

OLS regression in R that uses the Newton-Raphson algorithm

```
Raphsommodel=glm(y~1,data=data2,family=gaussian(link = "identity"))
summary(Raphsommodel)
```

```
##
## Call:
## glm(formula = y ~ 1, family = gaussian(link = "identity"), data = data2)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -12.0963   -6.1832   -0.7485    5.7643   15.2275
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  14.1616     0.5176   27.36   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 53.58635)
##
##     Null deviance: 10664  on 199  degrees of freedom
## Residual deviance: 10664  on 199  degrees of freedom
## AIC: 1366.8
##
## Number of Fisher Scoring iterations: 2
```

Linear model

```
linearmodel=lm(y~1,data=data2)
summary(linearmodel)
```

```
##
## Call:
## lm(formula = y ~ 1, data = data2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.0963  -6.1832  -0.7485   5.7643  15.2275
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  14.1616     0.5176   27.36   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.32 on 199 degrees of freedom
```

Since the OLS estimates for both Newton-Raphson algorithm and using lm is the two models are the same.