# Problem Set 1

Applied Stats II

Due: February 11, 2024

## Instructions

- Please show your work! You may lose points by simply writing in the answer. If the problem requires you to execute commands in `R`, please include the code you used to get your answers. Please also include the `.R` file that contains your code. If you are not sure if work needs to be shown for a particular problem, please ask.
- Your homework should be submitted electronically on GitHub in `.pdf` form.
- This problem set is due before 23:59 on Sunday February 11, 2024. No late assignments will be accepted.

## 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:

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

where $F$ is the theoretical cumulative distribution of the distribution being tested and $F_{(i)}$ is the $i$th 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:

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

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).

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 statistic
  D <- max(abs(empiricalCDF - pnorm(data)))
```

**The R script is:**

```
set.seed(123)
n <- 1000
emp <- rcauchy(n, location = 0, scale = 1)
# create a function that takes observed data as input
ks <- function(data) {
  # create empirical distribution of observed data
  ECDF <- ecdf(data)
  empiricalCDF <- ECDF(data)

  # generate test statistic
  D <- max(abs(empiricalCDF - pnorm(data)))

  addition <- 0
  for(a in 1:n){
    addition <- c(addition, exp((- (2 * a - 1)^2 * pi^2) / ((8 * D)^2)))
  }

  p <- sqrt(2 * pi) / D * sum(addition)

  print(paste("D =", D))
  print(paste("p_value =", p))
}

print(ks(emp))

ks.test(emp, "pnorm")
```

Dataset consists of rcauchy random variables and these random variables are entered into the defined ks function to derive the test statistic and p value through running loop several times which are as follows:
D = 0.13472806160635
p-value = 0.00380152787349343

The test statistic and p value may also be calculated by ks.test function which gives the following result:
D = 0.13573
p-value = 2.22e-16.

# 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 (123)
data <- data.frame(x = runif(200, 1, 10))
data$y <- 0 + 2.75*data$x + rnorm(200, 0, 1.5)
```

**The R script is:**

```
# Log-likelihood function for OLS regression
log_lkhd_ols <- function(beta, x, y) {
  y_pred <- beta[1] + beta[2] * x
  residuals <- y - y_pred
  log_lkhd_values <- dnorm(residuals, mean = 0, sd = 1.5, log = TRUE)
  return(-sum(log_lkhd_values))
}

# Use BFGS optimization to estimate OLS coefficients
initial <- c(0, 1)
result <- optim(par = initial, log_lkhd_ols, x = data$x, y = data$y, method =
    "BFGS")

ols_coef_log_lkhd <- result$par

print(paste("Intercept:", ols_coef_log_lkhd[1]))
print(paste("Slope:", ols_coef_log_lkhd[2]))

# Compare with lm function
lm_result <- lm(y ~ x, data = data)

ols_coef_lm <- coef(lm_result)

print(paste("Intercept:", ols_coef_lm[1]))
print(paste("Slope:", ols_coef_lm[2]))
```

The values of intercept and slope are similar i.e. 0.14 and 2.73 respectively (after rounding) using both BFGS and lm. Log likelihood method that use the dnorm function is used to estimate the coefficients, independent/dependent variables and residuals assuming normal distribution having a mean value of 0 and standard deviation of 1.5 to find the best model fit for the randomly generated data.