Problem Set 1

Applied Stats II

Due: February 14, 2022

Name: Gareth Moen

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 class on Monday February 14, 2022. No late assignments will be accepted.
- Total available points for this homework is 80.

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 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:

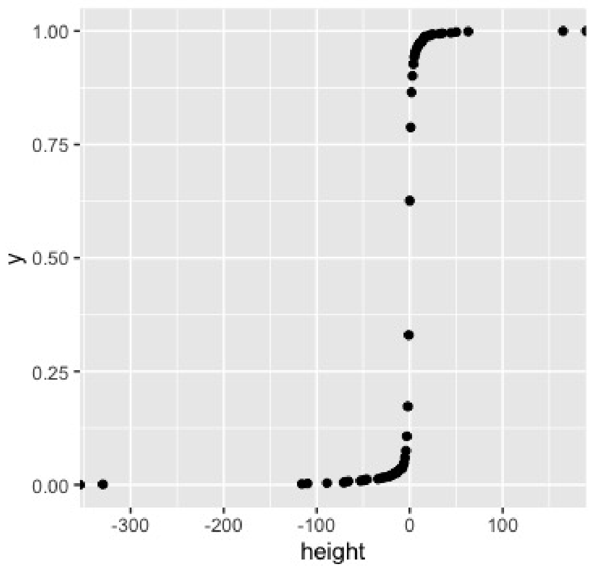
$$p(D \leq x) \frac{\sqrt{2\pi}}{x} \sum_{k=1}^{\infty} e^{-(2k-1)^2\pi^2/(8x^2)}$$

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).

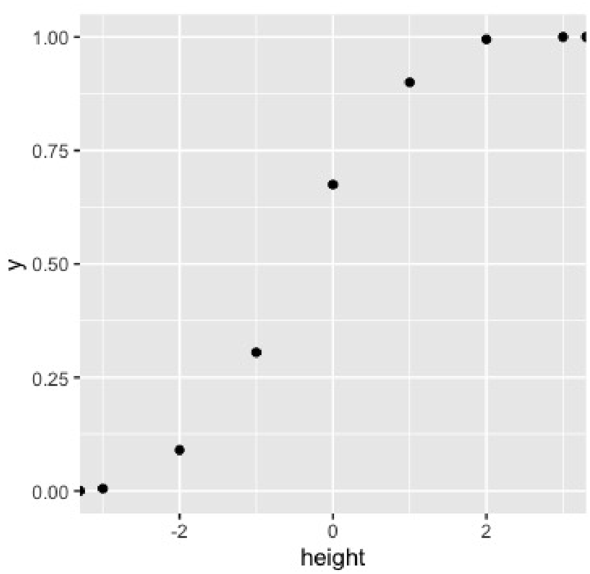
Write an `R` function that implements this test where the reference distribution is normal. As a hint, you can create the empirical distribution and theoretical CDF using this code:

```r
# Problem 1
#####################
# Note 1: We must compare the model CDF of the normal distribution to the cauchy
# empirical CDF to see if the distributions match
set.seed(1234)

data_1 <- (rcauchy(1000, location = 0, scale = 1)) # dataset from the cauchy distribution
empirical <- rnorm(1000, mean = 0, sd = 1) # dataset from the normal distribution for reference

#plotting the distributions on a chart
df_g1 <- data.frame(height = round(rcauchy(1000, location = 0, scale = 1)))
head(df)
df_g2 <- data.frame(height = round(rnorm(200, mean = 0, sd = 1)))
head(df)

ggplot(df_g1, aes(height)) + stat_ecdf(geom = "point") # See "Plot 1" below
ggplot(df_g2, aes(height)) + stat_ecdf(geom = "point") # See "Plot 2" below

ks.test(data_1, "pnorm") # built-in test comparing data_1 with normal distribution
# Results D = 0.14802
# p-value < 0.00000000000000022

#Function calculating the same values
Kol_Smir <- function(data_1, empirical) {
  n = 1000 # set the desired sum limit
  ECDF <- ecdf(data_1)
  empiricalCDF <- ECDF(data_1) # ecdf from rauchy distribution
  D <- max(abs(empiricalCDF - pnorm(empirical))) # max value of difference in ECDFs
  summed <- NULL #try to convert the D-value to a p-value and print it
    for(i in 1:33){
    summed <- c(summed, exp((-(2*i - 1)^2*pi^2) / ((8*D)^2)))
    }
  pValue <- sqrt(2*pi)/D*sum(summed)
```

```
  print(pValue)
}
print(D)

# With a p-value of less than 0.05 we can reject te null hypothesis and say
    that our
# sample data doesn't come from the normal distribution
```



Plot 1



Plot 2

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.

```r
######################
# Problem 2
######################

set.seed(1234)
x = runif(200, 1, 10)
data <- data.frame(x = runif(200, 1, 10))
data$y <- 0 + 2.75*data$x + rnorm(200, 0, 1.5) # intercept + slope*x + noise
pdf("data_dist.pdf")
plot(data$x, data$y)
dev.off()

# OLS regression with Newton-Raphson algorithm (BFGS)

# derive our log-likelihood function for uniform distribution
uniform_likelihood <- function(outcome, input, parameter) {
  p <- exp(parameter[1] + parameter[2]*input)/(1+exp(parameter[1] + parameter
    [2]*input))
  -sum(dunif(outcome, 1, p, log=TRUE))
}
#optimise our log-likelihood function
results_uniform <- optim(fn=uniform_likelihood, outcome=data$y, input=x, par
    =0:1, hessian=T, method="BFGS")

#coefficients
results_uniform$par

#lm function to confirm equivalent results
coef(lm(y ~ x, data=data))
# Intercept of 0.2956 and x value = 2.7221
#
```