Non-Parametric Methods

Amir Voloshin

12/10/2022

Libraries

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(readr)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.4.0     v purrr   0.3.4
## v tibble  3.1.5     v stringr 1.4.0
## v tidyr   1.2.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(xml2)
library(nortest)
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(KernSmooth)
```

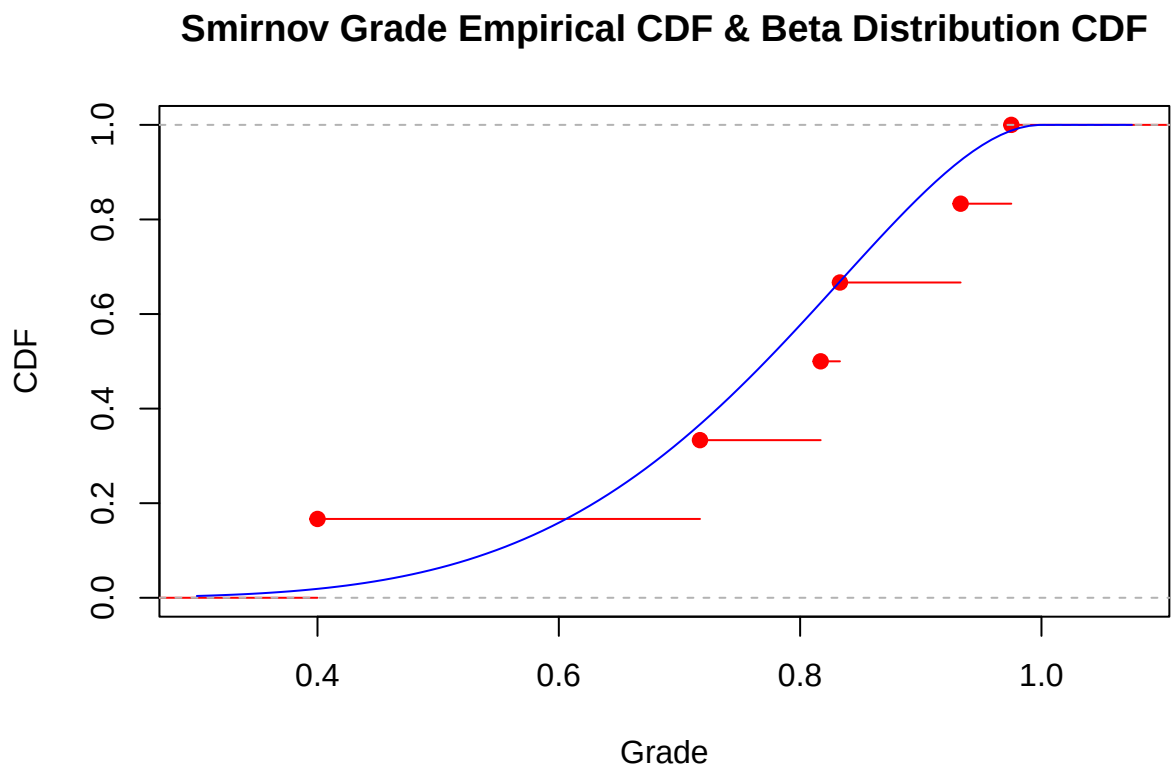
```
## KernSmooth 2.23 loaded
## Copyright M. P. Wand 1997-2009
```

Analyzing Empirical Data with Kolmogorov-Smirnov Test's and Density Curves

Problem 1

Part a

```
# Beta Plot
x <- seq(0, 1, length = 100)

# Smirnov Grades
grades <- c(0.40, 0.717, 0.817, 0.833, 0.933, 0.975)
emp_cdf <- ecdf(grades)
plot(emp_cdf, ylab = "CDF", xlab = "Grade", col = "red", main =
       "Smirnov Grade Empirical CDF & Beta Distribution CDF")
curve(pbeta(x, 6, 2), add = TRUE, col = "blue", main = "Beta Distribution CDF")
```

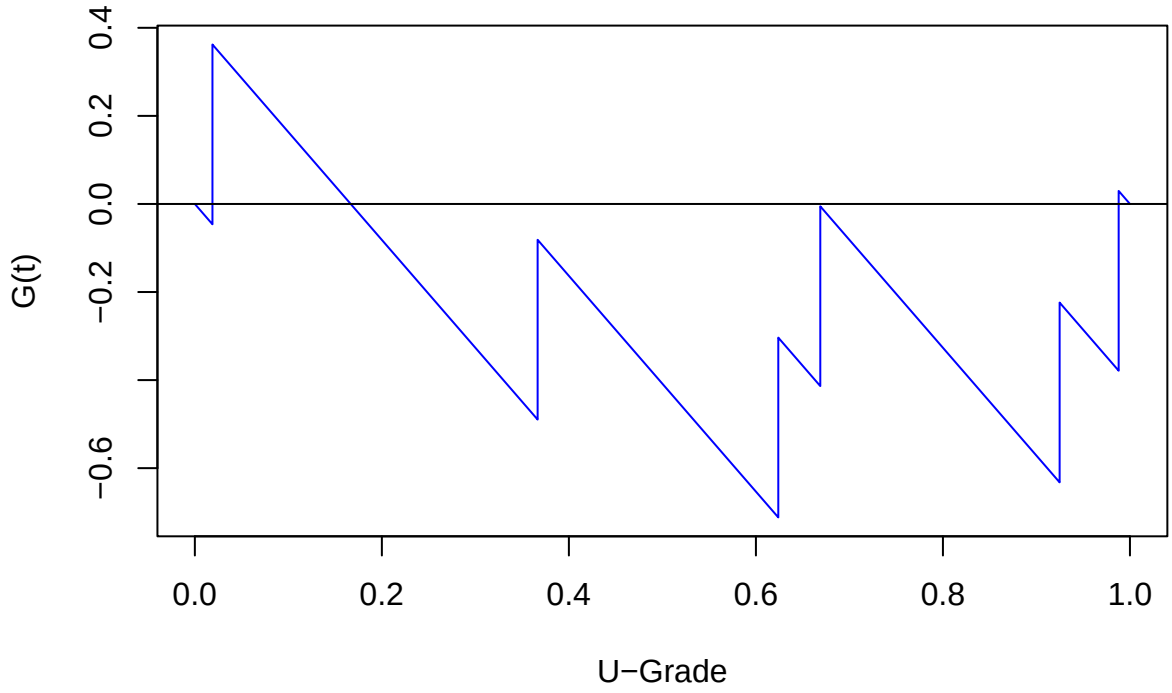


Part b

```r
# Following code from TA
u_grades <- pbeta(grades, 6, 2)
u_grades
```

```
## [1] 0.0188416 0.3665700 0.6239328 0.6688582 0.9247788 0.9879285
```

```r
plot(c(0, rep(u_grades, each = 2), 1), sqrt(6)*rep(seq(0, 1, by = 1/6), each = 2) - sqrt(6)*c(0, rep(u_
abline(h = 0)
```



Part c

```r
ks.test(grades, "pbeta", 6, 2)
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  grades
## D = 0.2906, p-value = 0.5955
## alternative hypothesis: two-sided
```

Part d
H_0: grades follow β(6,2)
H_A: grades do not follow β(6,2)
With a p-value of 0.5955, I do not reject the null hypothesis, and I can conclude that the grades likely do follow a beta distribution with alpha = 6 and beta = 2 parameters.

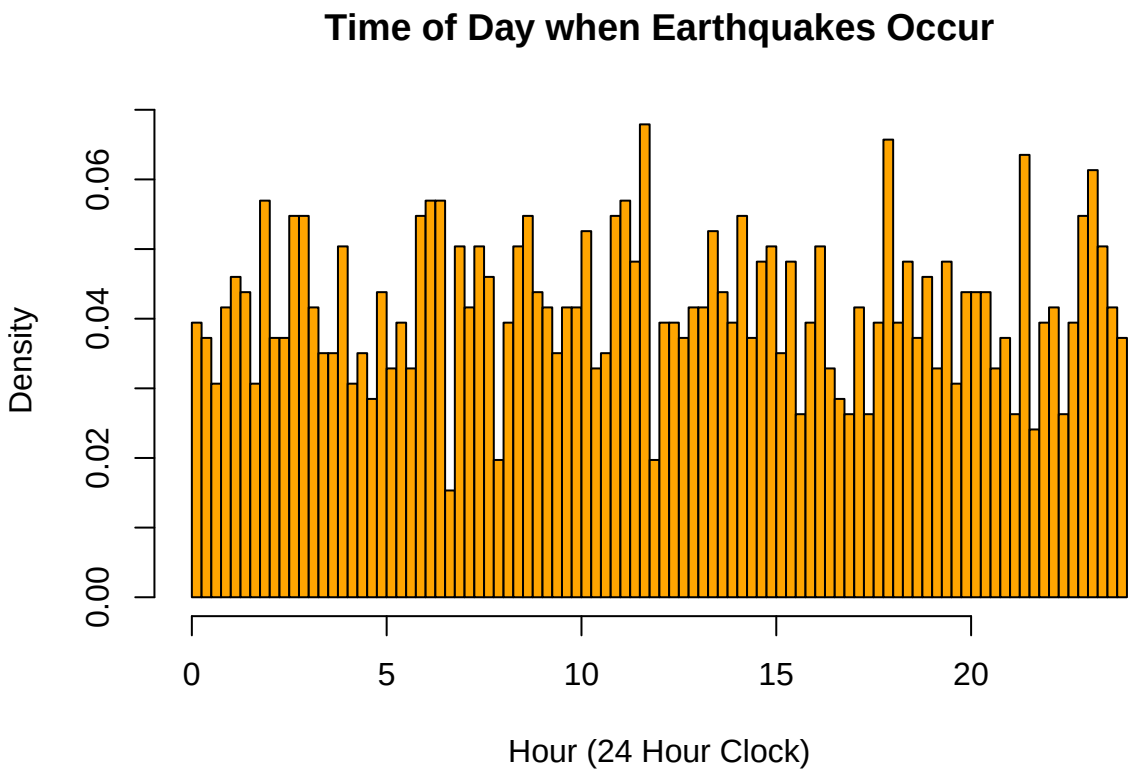
Part e
It would be very hard to perform a chi-squared test to test if this data follows a beta distribution because the chi-squared test is designed to test for discrete distributions, and the beta distribution is a continuous distribution.

Problem 2

```
quake <- read.table("EarthquakeData.htm", sep = ",", skip = 13, header = TRUE, nrows = 1826,
            colClasse = c("integer", "integer", "integer", "character", "numeric", "numeric",
                          "numeric", "integer", "factor"))
q_hour <- as.integer(substr(quake$Time.hhmmss.mm.UTC, 1, 2))
q_min <- as.integer(substr(quake$Time.hhmmss.mm.UTC, 3, 4))
q_sec <- as.numeric(substr(quake$Time.hhmmss.mm.UTC, 5, 8))
# quake
```

Part a

```
# Transforming time into hours after start of a day
time_hour <- (q_hour*60 + q_min + q_sec/(60))/60

# 15 minute widths
width <- seq(0, 24, by = 1/4)

hist(time_hour, breaks = width, probability = TRUE,
     main = "Time of Day when Earthquakes Occur", xlab = "Hour (24 Hour Clock)"
     , col = "orange")
```



Part b

```
t_hour <- table(q_hour)
exp <- sum(t_hour)/24

# Chi-Squared Test
chi_stat <- sum((t_hour - exp)^2)/exp
chi_stat
```

```
## [1] 16.63746
```

```
# Obtaining the Critical Value
qchisq(0.95, 23)
```

```
## [1] 35.17246
```

H_0: Hour of Earthquakes are equally distributed among 24 hours
H_A: Hour of Earthquakes are not equally distributed among 24 hours
With a critical value of 35.17246, and a chi-squared value of 16.63746, we do not reject the null hypothesis, and conclude that the hour of day which earthquakes occur are likely equally distributed among 24 hours.

Part c

```
# Transforming time into fractions of a day (from TA)
time_mod <- q_hour/24 + q_min/(24*60) + q_sec/(24*60*60)
```

```r
# Following code from lecture
d_plus <- max(seq(1/1826, 1, by = 1/1826) - sort(time_mod))
d_minus <- max(sort(time_mod) - seq(0, 1825/1826, by = 1/1826))
D <- max(d_plus, d_minus)
D
```

```
## [1] 0.01338519
```

Part d

```r
# Brownian-Bridge Approximation
p <- 2*(exp((-2)*((sqrt(1826)*D)^2) - exp((-8)*((sqrt(1826)*D)^2)) + exp((-18)*((sqrt(1826)*D)^2))) )
p
```

```
## [1] 0.9690951
```

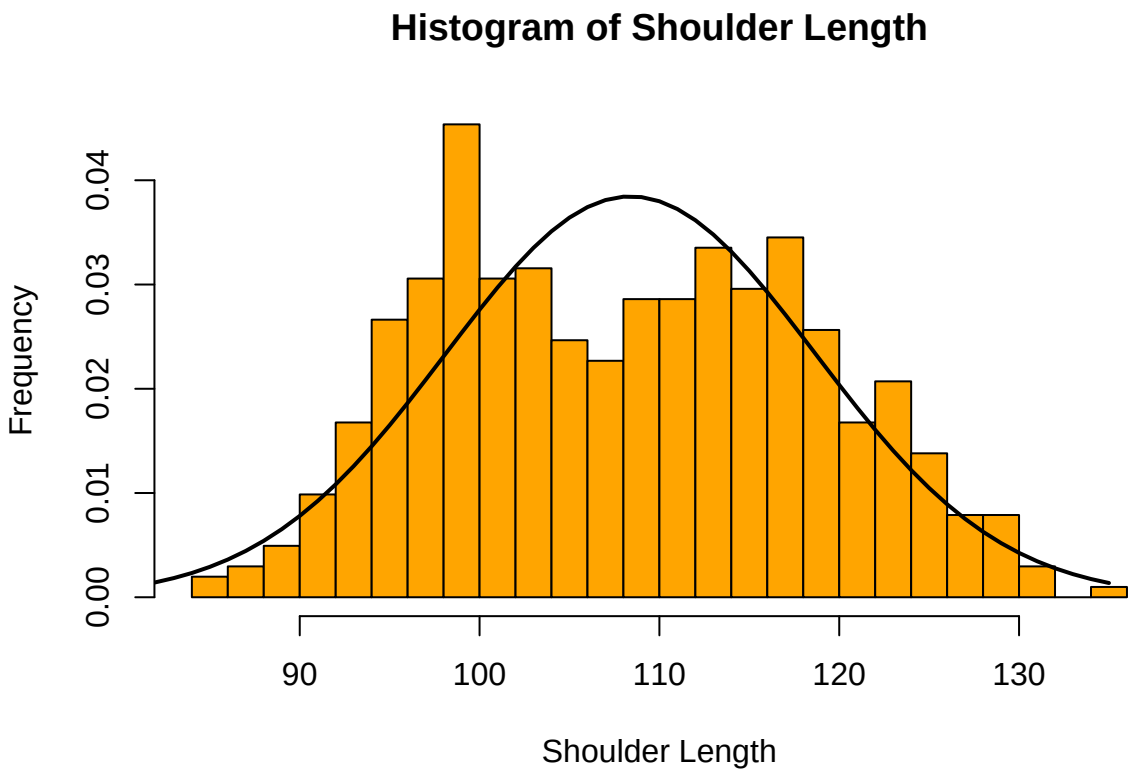
Approximated P-value = 0.9690951

Problem 3

```r
# Reading in the Data
shldr <- read.table("shoulder.txt", col.names = c("Shoulder", "Gender"), skip = 1)
```

Part a

```r
x = dnorm(x = seq(0, 135, length = 135), mean = 108.1951, sd = 10.37483)
hist(shldr$Shoulder, breaks = 20, main = "Histogram of Shoulder Length",
     prob = TRUE, xlab = "Shoulder Length", ylab = "Frequency", col = "orange")
lines(x, lwd = 2, col = "black")
```



Part b

```
lillie.test(shldr$Shoulder)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  shldr$Shoulder
## D = 0.077918, p-value = 8.838e-08
```

```
cvm.test(shldr$Shoulder)
```

```
##
##  Cramer-von Mises normality test
##
## data:  shldr$Shoulder
## W = 0.62429, p-value = 2.321e-07
```

```
ad.test(shldr$Shoulder)
```

```
##
##  Anderson-Darling normality test
##
## data:  shldr$Shoulder
## A = 3.6469, p-value = 4.112e-09
```

H_0: Shoulder Lengths follow a Normal Distribution
H_A: Shoulder Lengths do not follow a Normal Distribution
Each of these three tests for normality, Lilliefors, Cramer-von, and Anderson-Darling, resulted in very small p-values. So we reject the null hypothesis and conclude that the Shoulder Length data does not follow a normal distribution.

Part c

```
# Splitting the data
mshldr <- shldr %>%
          filter(shldr$Gender == 'Male')
fshldr <- shldr %>%
          filter(shldr$Gender == 'Female')

# Test for Male
lillie.test(mshldr$Shoulder)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  mshldr$Shoulder
## D = 0.041117, p-value = 0.3921
```

```
cvm.test(mshldr$Shoulder)
```

```
##
##  Cramer-von Mises normality test
##
## data:  mshldr$Shoulder
## W = 0.035563, p-value = 0.7605
```

```
ad.test(mshldr$Shoulder)
```

```
##
##  Anderson-Darling normality test
##
## data:  mshldr$Shoulder
## A = 0.2263, p-value = 0.8158
```

```
# Test for Female
lillie.test(fshldr$Shoulder)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  fshldr$Shoulder
## D = 0.076669, p-value = 0.0008286
```

```
cvm.test(fshldr$Shoulder)
```

```
##
```

```
##  Cramer-von Mises normality test
##
## data:  fshldr$Shoulder
## W = 0.28647, p-value = 0.0004713
```

```
ad.test(fshldr$Shoulder)
```

```
##
##  Anderson-Darling normality test
##
## data:  fshldr$Shoulder
## A = 1.6058, p-value = 0.0003889
```

After re-running the test for men and women, all the tests for men have large p-values, while all the tests for women have very small p-values. So I can conclude that it is very likely that the shoulder length of men does follow a normal distribution. I can also conclude that it is very likely that the shoulder length of women does not follow a normal distribution.

Part d

```
m_mu <- mean(mshldr$Shoulder)
f_mu <- mean(fshldr$Shoulder)

m_new <- mshldr$Shoulder - m_mu
f_new <- fshldr$Shoulder - f_mu
shldr_new <- c(m_new, f_new)

# Tests
lillie.test(shldr_new)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  shldr_new
## D = 0.041835, p-value = 0.03391
```

```
cvm.test(shldr_new)
```

```
##
##  Cramer-von Mises normality test
##
## data:  shldr_new
## W = 0.16851, p-value = 0.01357
```

```
ad.test(shldr_new)
```

```
##
##  Anderson-Darling normality test
##
## data:  shldr_new
## A = 0.94516, p-value = 0.01668
```

Each of the three normality tests still result in a small p-value at the 0.05 significance level. So I conclude that the shoulder length data set still does not follow a normal distribution.

Part e
I do not think it would be reasonable to assume normality and perform a two sample t-test. In part c I tested for normality for men and women separately, and my results were that only men shoulder length are normally distributed. So since I concluded that women shoulder lengths are not normally distributed, I cannot assum normality for a two sample t-test.

Problem 4

```
# Reading in the data
BasketBDays <- read_csv("Basketball Reference BDays.txt",
       col_types = cols(`Birth Date` = col_date(format = "%B %d %Y")))

BasketBDays$Month <- as.numeric(format(BasketBDays$`Birth Date`,
                            format="%m"))
BasketBDays$Year <- as.numeric(format(BasketBDays$`Birth Date`,
                            format="%Y"))
BasketBDays$Day <- as.numeric(format(BasketBDays$`Birth Date`,
                            format="%d"))
BasketBDays <- filter(BasketBDays, !is.na(Month))
# BasketBDays
```

```
# Adjusting data so each day is equal
days_month <- c(31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31)
days_so_far <- c(0, cumsum(days_month[1:11]))
BasketBDays <- mutate(BasketBDays, day_of_year = Day + days_so_far[Month],
                      prop_of_year = day_of_year/365)

# Taking into account leap year
BasketBDays <- mutate(BasketBDays, leap_year = (Year %% 4) == 0, day_of_year = Day +
                        days_so_far[Month] + leap_year*(Month > 2), prop_of_year = day_of_year/(365 + l

# Kolmogorov-Smirnov Test
ks.test(BasketBDays$prop_of_year, "punif", 0, 1)
```

```
## Warning in ks.test(BasketBDays$prop_of_year, "punif", 0, 1): ties should not be
## present for the Kolmogorov-Smirnov test
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  BasketBDays$prop_of_year
## D = 0.012744, p-value = 0.3816
## alternative hypothesis: two-sided
```

H_0: Basketball Births follow a Uniform Distribution
H_A: Basketball Births do not follow a Uniform Distribution
With a large p-value of 0.3816 we do not reject the null hypothesis. So I conclude that it is possible that the Basketball Births do follow a Uniform Distribution.

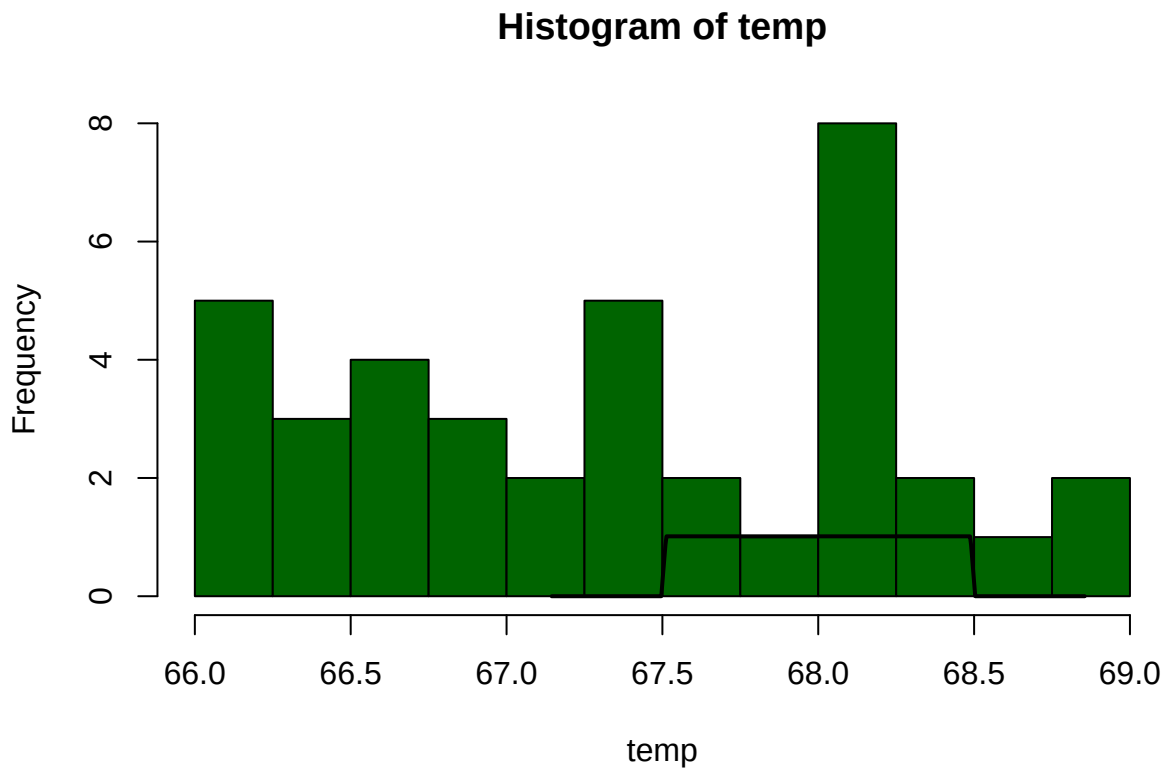
Kernel Density Estimation and Density Curves

Problem 5

```
# Loading the data
temp <- c(66.03, 66.04, 66.13, 66.14, 66.22, 66.32,
66.39, 66.47, 66.6, 66.63, 66.63, 66.66,
66.93, 66.99, 67, 67.22, 67.23, 67.31,
67.34, 67.42, 67.43, 67.48, 67.58, 67.67,
67.98, 68.02, 68.04, 68.13, 68.14, 68.15,
68.17, 68.19, 68.21, 68.36, 68.48, 68.69,
68.76, 68.92)
```

Part a

```
hist(temp, col = "dark green", probability = FALSE, breaks = seq(66, 69, by = .25))
lines(density(68, bw = .285, kernel = "rectangular", width = 1), lwd = 2)
```




```
# 13 obs between 67.5 and 68.5
n1 <- 500

# Density Estimate
f_hat <- 13/n1
f_hat
```

```
## [1] 0.026
```

My density estimate is 0.026

Part b

```
var_fhat <- (f_hat*(1 - f_hat))/(n1*(1^2))
var_fhat
```

```
## [1] 5.0648e-05
```

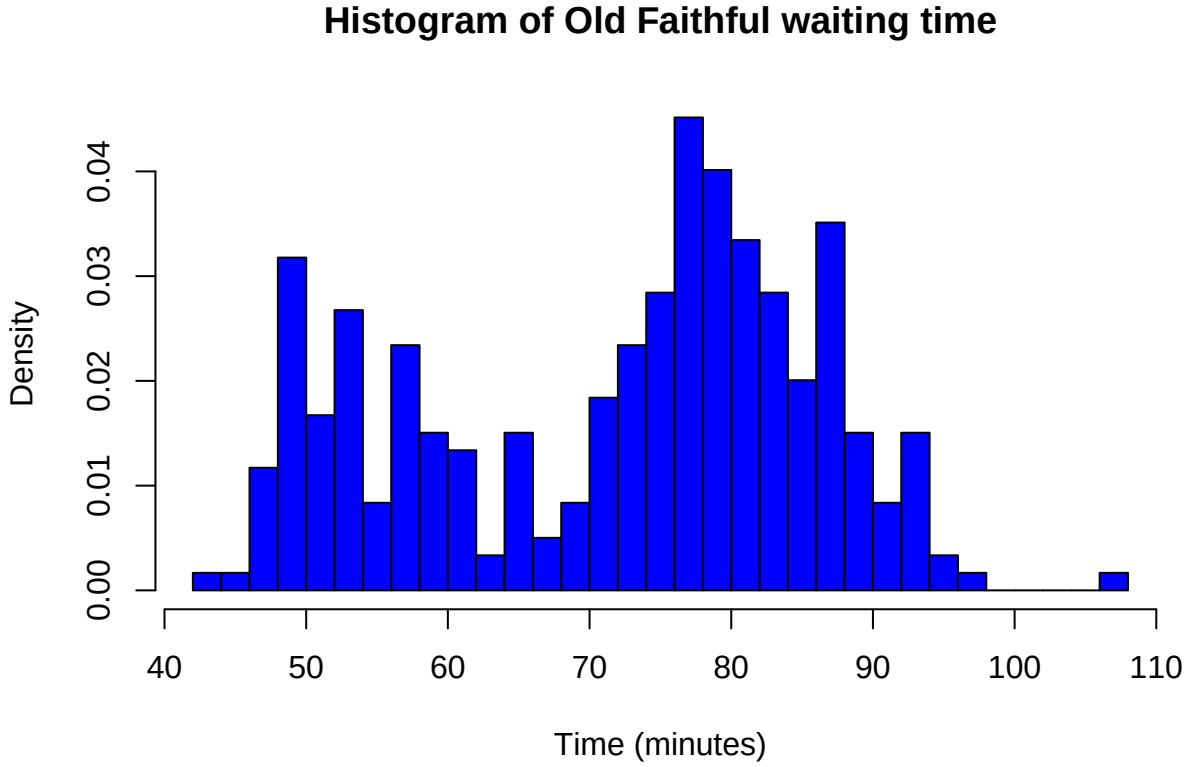
My estimated variance of f hat is 5.0648e-05

Part c
If we decide to use a Gaussian kernel over a rectangular kernel, we would have a more accurate estimate. We would have a more accurate estimate because a Gaussian kernel is a continuous kernel, and the temperature data is continuous, thus a Gaussian kernel will give us a smooth continuous kernel with less bias than a rectangular kernel. I would expect to use a smaller bandwidth, because when using a Gaussian kernel with a bandwidth of one, it replicates a normal distribution bell curve. If we are trying to achieve a kernel width of 1 (i.e. 67.5 to 68.5), we will need a smaller bandwidth than a rectangular kernel to achieve that interval.

Problem 6

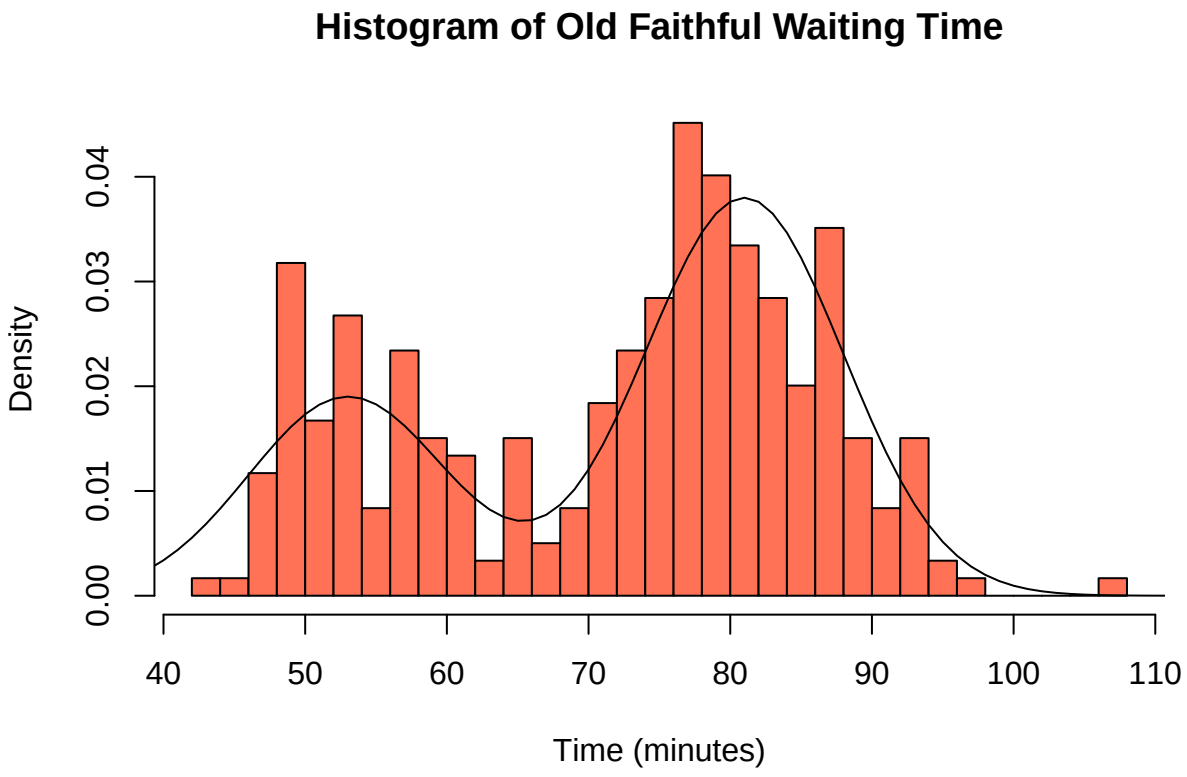
Part a

```
hist(geyser$waiting, col = "blue", probability = TRUE, breaks = 30, main = "Histogram of Old Faithful wa
```



Part b

```r
phi1 <- dnorm(0:1000, mean = 52, sd = 7)
phi2 <- dnorm(0:1000, mean = 80, sd = 7)
f <- (1/3)*phi1 + (2/3)*phi2

hist(geyser$waiting, col = "coral1", probability = TRUE, breaks = 30, main = "Histogram of Old Faithful
lines(f)
```



Part c

The given mixture density does look like it fits the data. The mixture density has two modes just like the data, at around the same data points.

Part d

```r
# following function from TA
mixcdf <- function(x){
  return(pnorm(x, 52, 7)/3 + 2*pnorm(x, 80, 7)/3)
}

# ks.test
ks.test(geyser$waiting, "mixcdf")
```

```
## Warning in ks.test(geyser$waiting, "mixcdf"): ties should not be present for the
## Kolmogorov-Smirnov test
```

```
##
```

```
##  One-sample Kolmogorov-Smirnov test
##
## data:  geyser$waiting
## D = 0.081275, p-value = 0.0385
## alternative hypothesis: two-sided
```

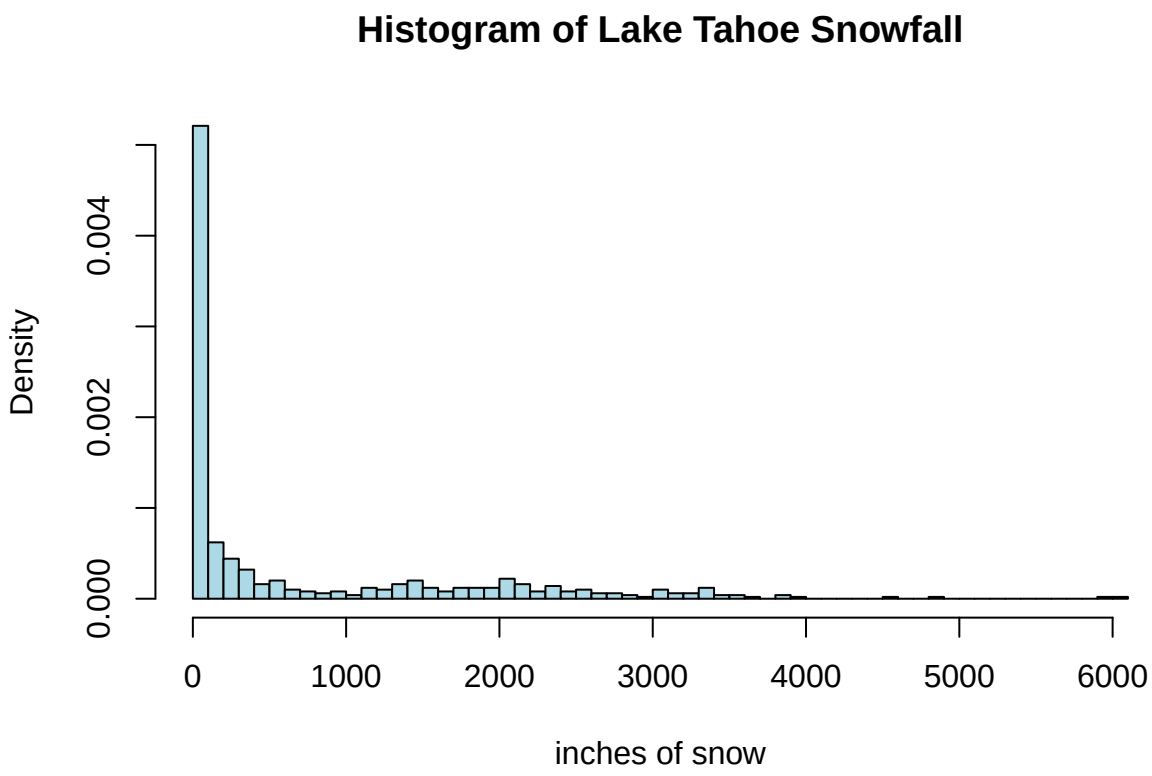
My Kolmogorov Smirnov test resulted in a p-value of 0.0385, so at the 0.05 significance level, I reject the null hypothesis that the geyser waiting time fits the mixture density. Thus I conclude that the mixture density f(x) does not fit the geyser data.

Problem 7

```
# Loading the data
snow <- scan("snow.txt")
```

Part a

```
hist(snow, col = "light blue", probability = TRUE, breaks = 50, main = "Histogram of Lake Tahoe Snowfall
```



Part b

```
# Bandwitdh of 200
h <- 200
f_hat4 <- mean(snow > (2000 - h/2) & snow < (2000 + h/2))/h
f_hat4
```

```
## [1] 0.0001703407
```

```
# 95% CI for f hat
p_hat <- h*f_hat4
CI_upper <- (p_hat + 1.96*sqrt(p_hat*(1-p_hat)))/h
CI_lower <- (p_hat - 1.96*sqrt(p_hat*(1-p_hat)))/h
CI <- c(CI_lower, CI_upper)
CI
```

```
## [1] -0.001607421  0.001948102
```

My density estimate for 2000 is 0.0001703407, using a bandwidth of 200.
My 95% Confidence Interval for this density estimate is (-0.001607421, 0.001948102).
I used a bandwidth of 200 because I it covers a sufficient amount of observations around 2000 to produce a good estimate of the density.

Part c

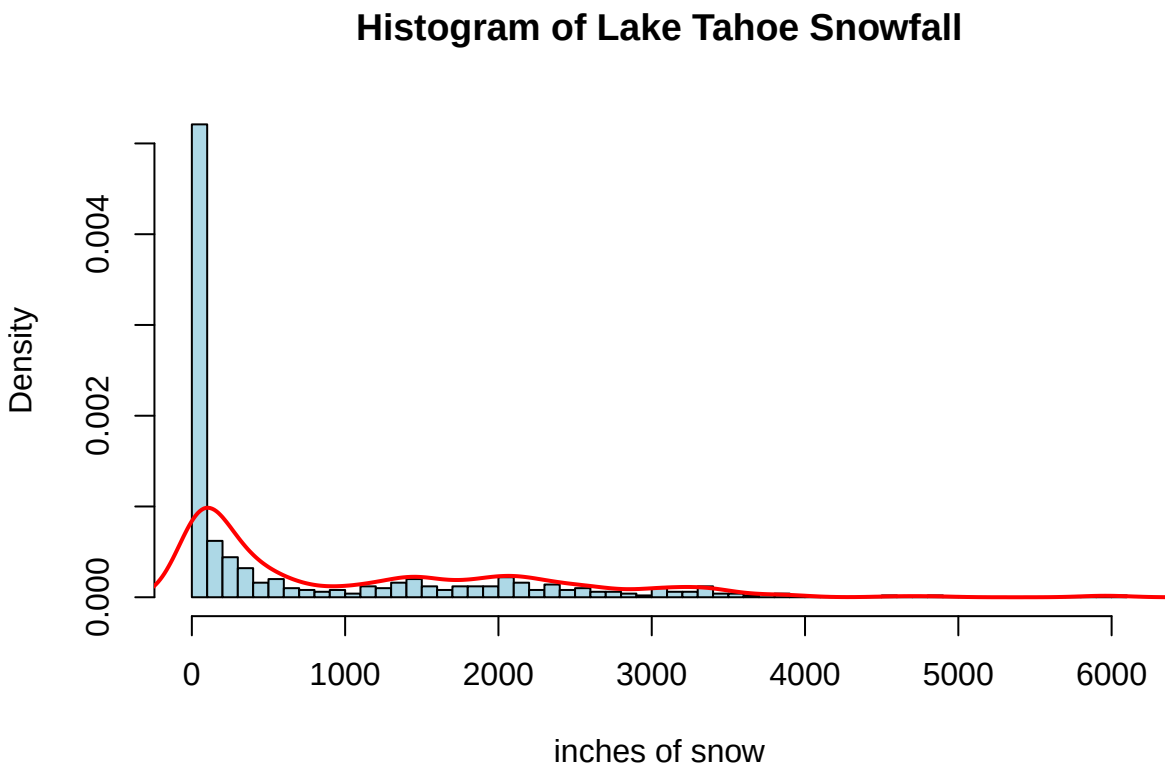
```
# Probability that a station sees no snow
# 183 stations with no snow
length(snow[snow == 0])/length(snow)
```

```
## [1] 0.3667335
```

The estimated probability that a station does not see snow is 0.3667335

Part d

```
# Getting rid of zero values
snow_df <- data.frame(snow)
snow_df[snow_df == 0] <- NA
snow_mod <- snow_df[complete.cases(snow_df),]

# Histogram
hist(snow, col = "light blue", probability = TRUE, breaks = 50, main = "Histogram of Lake Tahoe Snowfal
lines(density(snow_mod, bw = 150, kernel = "gaussian"), lwd = 2,
     col = "red")
```



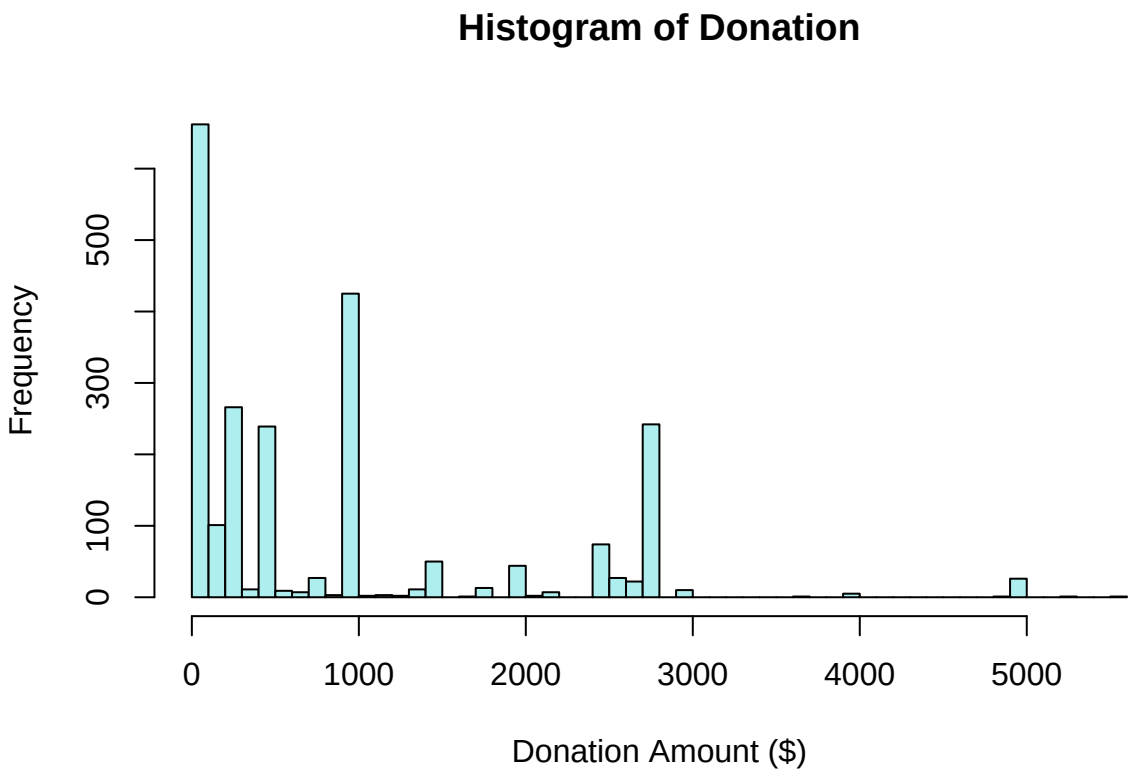
Problem 8

```
# Loading the data
carb <- read_csv("Carbajal.txt")
```

```
## Rows: 2295 Columns: 1
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (1): Donation
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

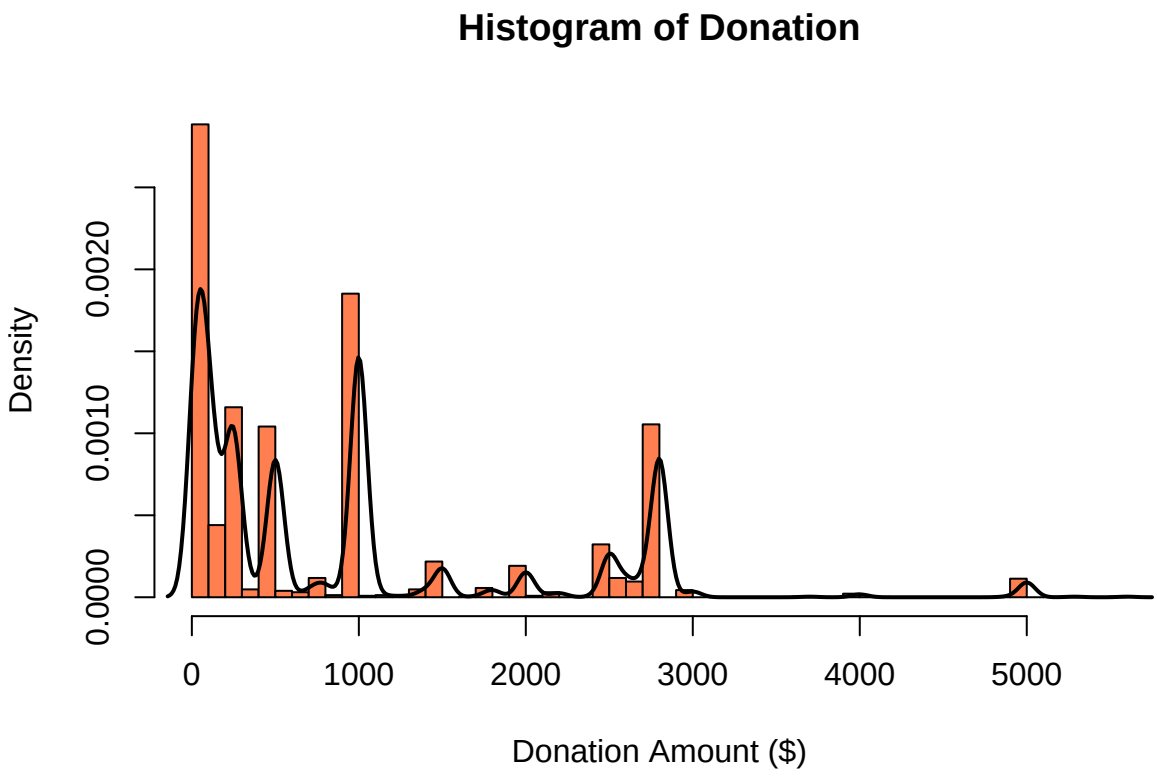
Part a

```
hist(carb$Donation, breaks =  40, col = "paleturquoise", main = 'Histogram of Donation', xlab = "Donati
```



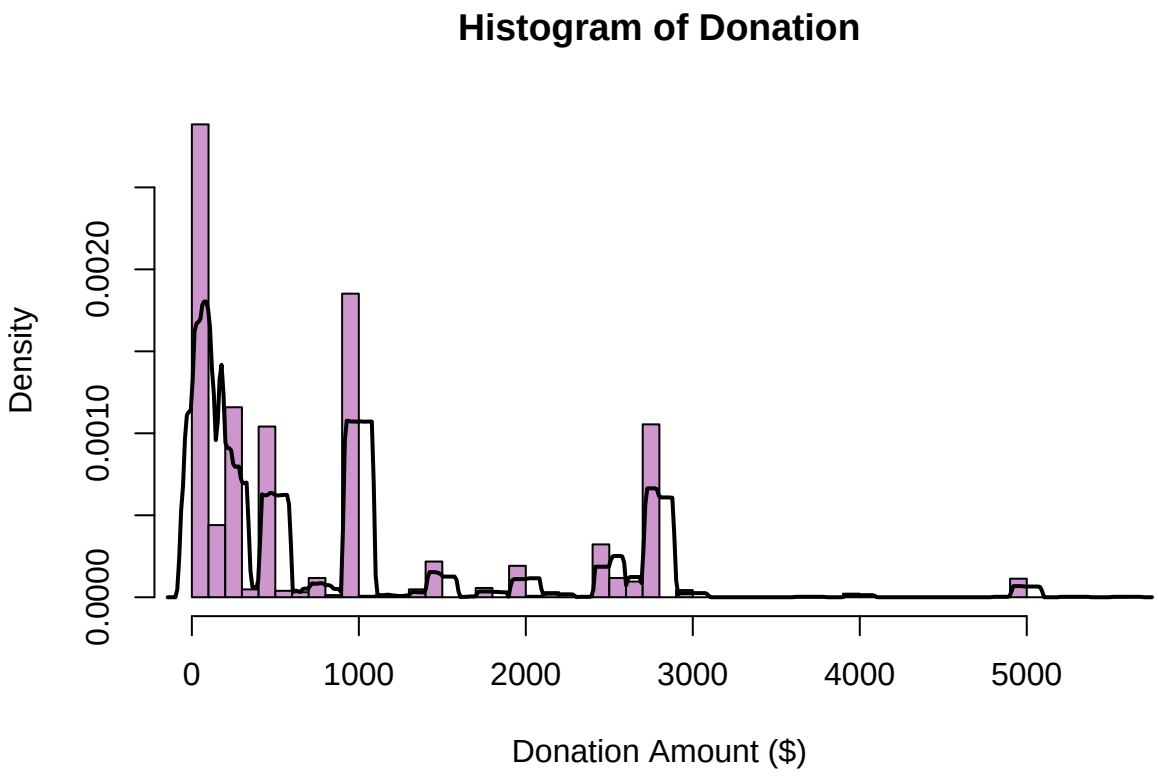
Part b

```
# h = 50, Gaussian kernel
hist(carb$Donation, breaks =  40, probability = TRUE, col = "coral", main = 'Histogram of Donation', xla
lines(density(carb$Donation, bw = 50, kernel = "gaussian"), lwd = 2,
     col = "black")
```

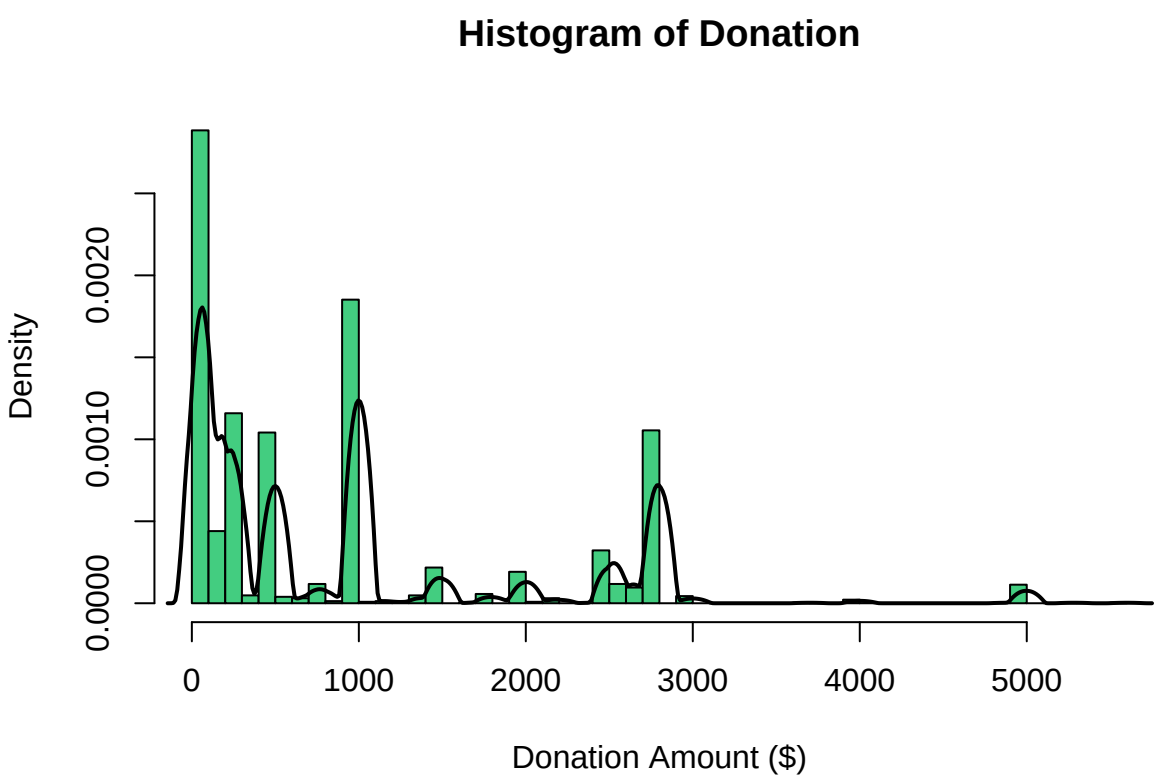


Part c

```
# Rectangular Kernel
hist(carb$Donation, breaks = 40, probability = TRUE, col = "plum3", main = 'Histogram of Donation', xlab
lines(density(carb$Donation, bw = 50, kernel = "rectangular"), lwd = 2,
     col = "black")
```




```
# Epanechnikov Kernel
hist(carb$Donation, breaks =  40, probability = TRUE, col = "seagreen3", main = 'Histogram of Donation'
lines(density(carb$Donation, bw = 50, kernel = "epanechnikov"), lwd = 2,
     col = "black")
```

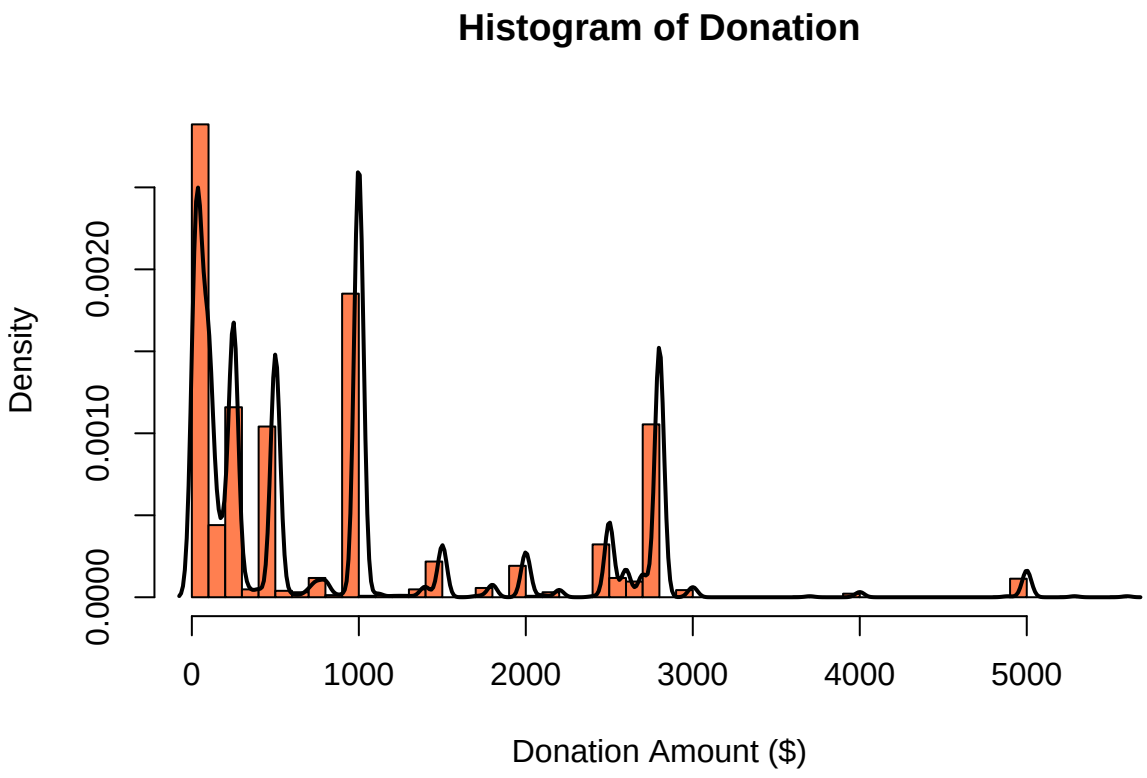


Part d
Between the rectangular and epanechnikov kernels for this histogram, the rectangular kernel, as it suggests, has sharper lines because of the edges. On the other hand the epanechnikov kernel really smooths out those edges to make a nicer curve.
Furthermore, because of the rectangular kernels edges, its derivative is not a continuous function, while the epanechnikov is because of the curve.

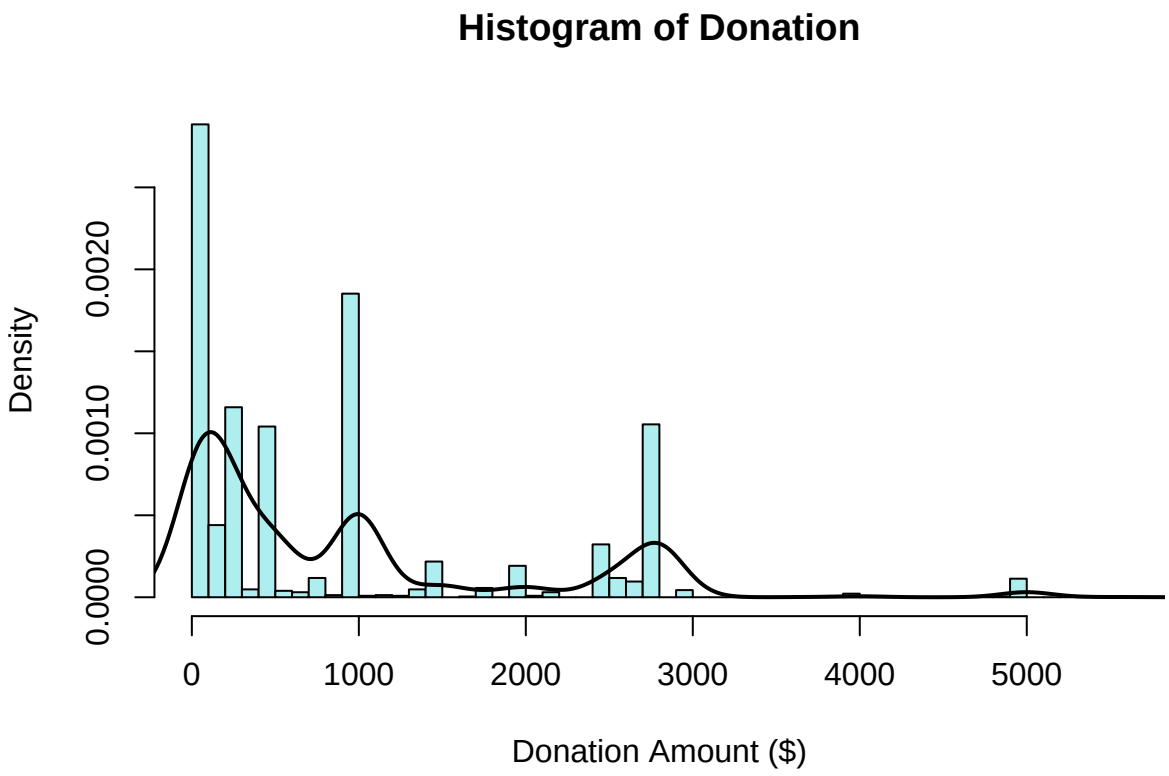
Part e

```
# ucv
hist(carb$Donation, breaks =  40, probability = TRUE, col = "coral", main = 'Histogram of Donation', xla
lines(density(carb$Donation, bw = "ucv", kernel = "gaussian"), lwd = 2,
     col = "black")
```

```
## Warning in bw.ucv(x): minimum occurred at one end of the range
```




```
# nrd
hist(carb$Donation, breaks =  40, probability = TRUE, col = "paleturquoise", main = 'Histogram of Donati
lines(density(carb$Donation, bw = "nrd", kernel = "gaussian"), lwd = 2,
     col = "black")
```



Part f

```
# Density Estimate at $50
h1 <- 200
f_50 <- mean(carb$Donation > (50 - h1/2) & carb$Donation < (50 + h1/2))/h1
f_50
```

```
## [1] 0.001455338
```

```
# Density Estimate at $1500
h2 <- 200
f_1500 <- mean(carb$Donation > (1500 - h2/2) & carb$Donation < (1500 + h2/2))/h2
f_1500
```

```
## [1] 0.0001089325
```

My density estimate with a bandwidth of 200 at $50 is 0.001455338
At $1500 with a bandwidth of 200 my density estimate is 0.0001089325

Part g
The bandwidth has a great effect on the resulting density estimate than the kernel shape. This is because the bandwidth determines how many observations you grab to the left and right of the value you are trying to estimate, and all those values can significantly change the density estimate. So the bandwidth has more weight in determining the density estimate than the shape.

Local Linear and Nadayara Watson Estimators

Problem 9

```
# Reading in the data
andro <- read.table("andro2.txt", header = TRUE)
```

Part a

```
# Estimate of mean at time 200
index <- which(andro$Time == 200)
x_bar1 <- mean(andro$Signal[(index - 17):(index + 17)])
x_bar1
```

```
## [1] 0.02257143
```

```
# Calculations for CI
s1 <- sd(andro$Time > (index - 17) & andro$Time < (index + 17))
n1 <- 35
df1 <- 35 - 2
t_star1 <- qt(c(0.025, 0.975), df1)
se1 <- sd(andro$Signal[index+(-17:17)])/sqrt(n1)

# 95% CI for mean
x_bar1 + t_star1*se1
```

```
## [1] -0.01242318  0.05756603
```

My estimated value of the mean function at time 200 with the closest 35 points is 0.02257143
My 95% CI for this estimate is (-0.01242318, 0.05756603)

Part b

```
# Estimated mean
x_bar2 <- mean(andro$Signal[(index - 52): (index + 52)])
x_bar2
```

```
## [1] -0.03104762
```

```
# Calculations for CI
n2 <- 105
df2 <- 105 - 2
t_star2 <- qt(c(0.025, 0.975), df2)
se2 <- sd(andro$Signal[index+(-52:52)])/sqrt(n2)

# 95% CI for mean
x_bar2 + t_star2*se2
```

```
## [1] -0.04493442 -0.01716082
```

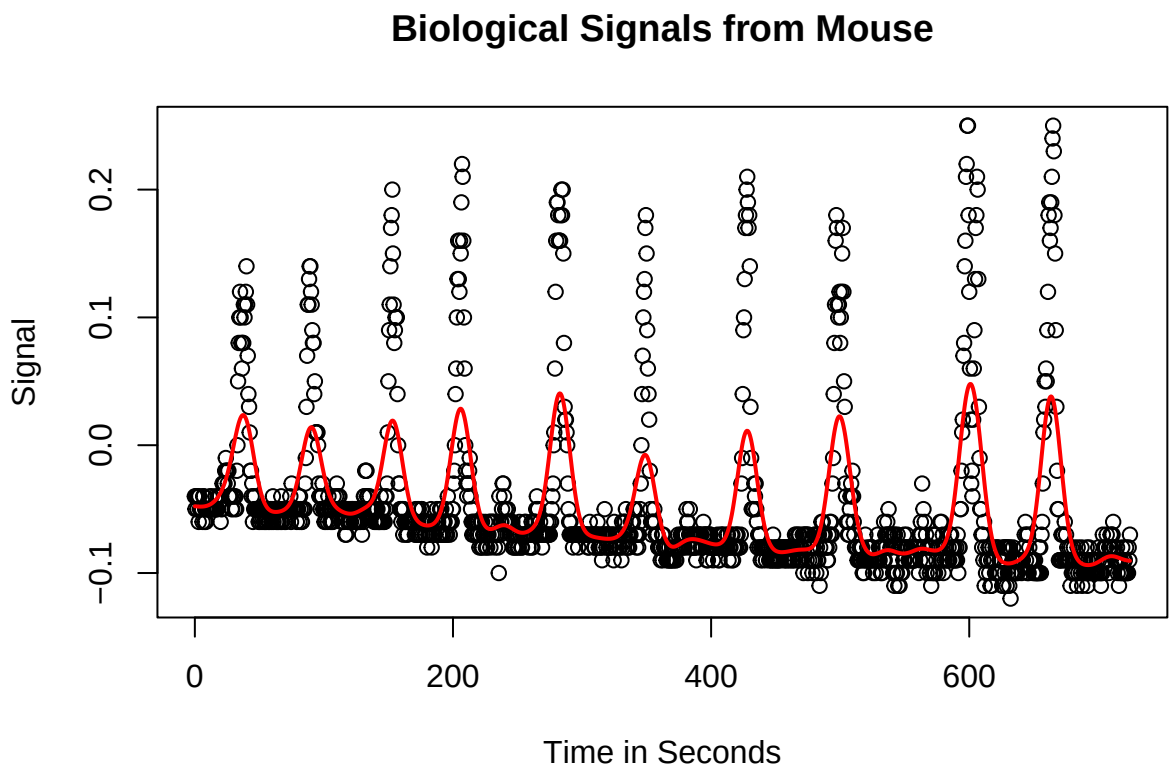
My estimated value of the mean function at time 200 with the 105 closest points is -0.03104762
My 95% CI for this estimate is (-0.04493442, -0.01716082)

Part c

The bias of the estimate is not accounted for in my 95% confidence interval from part a because for bias we consider the true mean, but we are not taking the true mean into consideration at all here. In part a we are taking a local mean, and thus bias is not accounted for in my confidence interval.

Part d

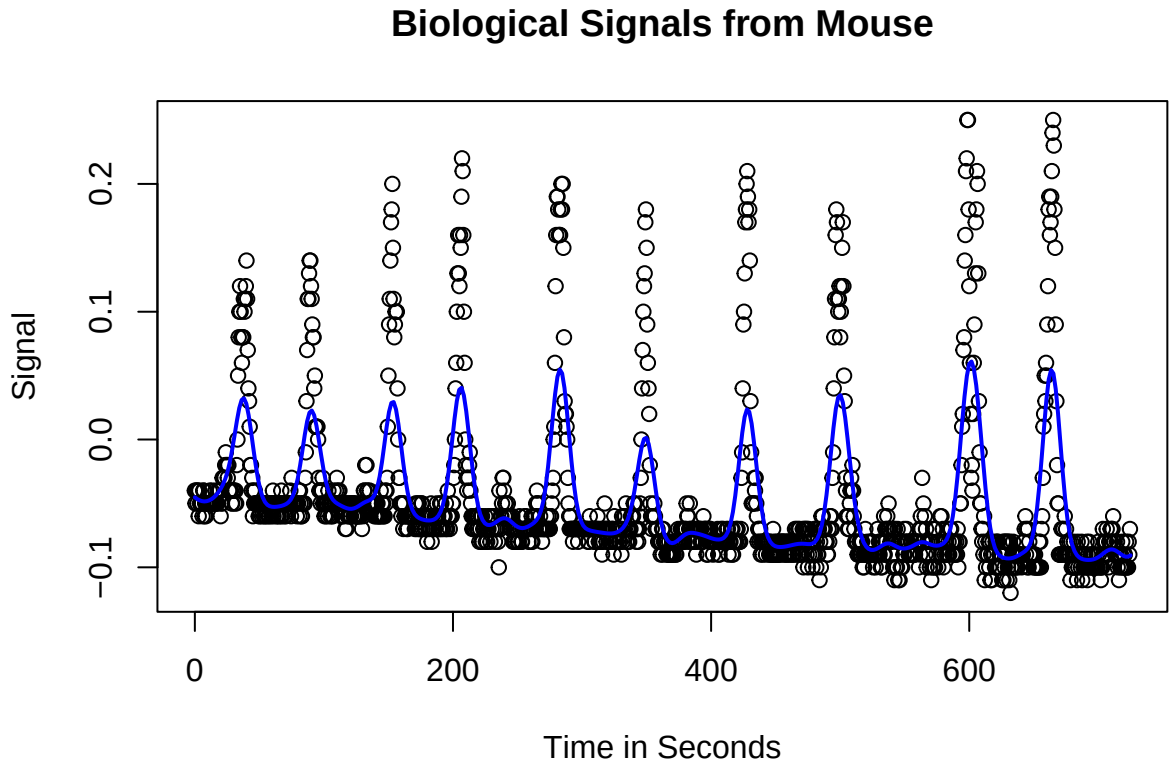
```
# Following code from lecture
# Function for Nadaraya-Watson
NWfit <- function(t, x, y, bw){
  GausK <- dnorm(x, mean = t, sd = bw)
  sum(y*GausK)/sum(GausK)
}

# Plotting
NW.hat <- sapply(andro$Time, NWfit, x = andro$Time, y = andro$Signal, bw = 7)
plot(x = andro$Time, y = andro$Signal, main = "Biological Signals from Mouse", ylab = "Signal", xlab =
lines(andro$Time, NW.hat, type = "l", col = "red", ylab = "NW", lwd = 2)
```



Part e

```
# Local-linear Regression fit
x1 <- seq(0, 725, length = 1450)
reg.est.andro.ll <- locpoly(x = x1 , y = andro$Signal,
                            degree = 1, bandwidth = 6)
```

```r
plot(x = andro$Time, y = andro$Signal, main = "Biological Signals from Mouse", ylab = "Signal", xlab = "
lines(reg.est.andro.ll, col = "blue", lwd = 2)
```



Part f

```r
# Following code from TA
m <- length(reg.est.andro.ll$y)
peaks <- (reg.est.andro.ll$y[2:(m-1)] > reg.est.andro.ll$y[1:(m-2)] & reg.est.andro.ll$y[2:(m-1)] > reg

# Local peak values
peaks_big <- peaks & (reg.est.andro.ll$y[2:(m-1)] > 0)
sum(peaks_big)
```

```
## [1] 10
```

```r
# Mean time between peaks
spaces <- diff(reg.est.andro.ll$x[peaks_big])
mean(spaces)
```

```
## [1] 69.47917
```

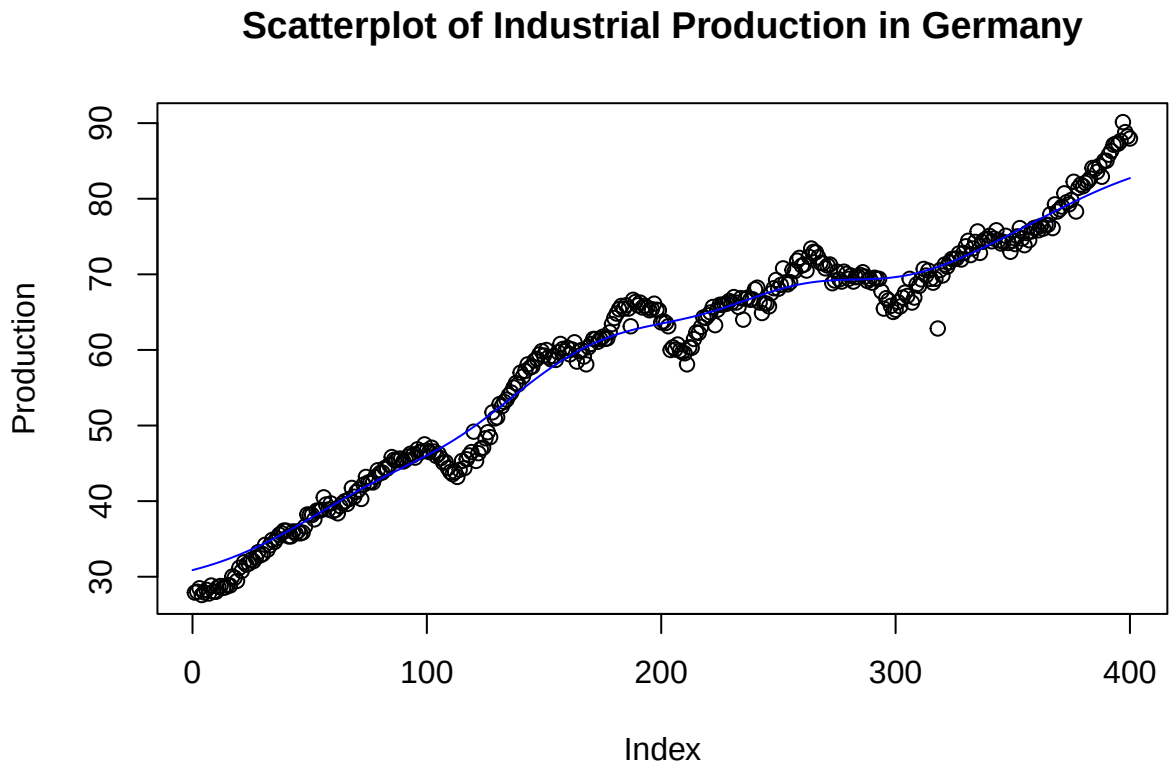
Using my estimates from the mean function, I found that there are 10 peaks in this sequence of data, with each peak occurring on average every 69.47917 seconds.

Problem 10

```
# Reading in the data
germ <- read.table("GermProd.txt", header = TRUE)
```

Part a

```
# Nadaraya-Watson
x <- seq(0, 400, length = 400)
reg.estimate.NW <- locpoly(x = x, y = germ$production,
                           degree = 0, bandwidth = 24)
plot(germ$production, main = "Scatterplot of Industrial Production in Germany", xlab = "Index", ylab =
lines(reg.estimate.NW, col = "blue")
```

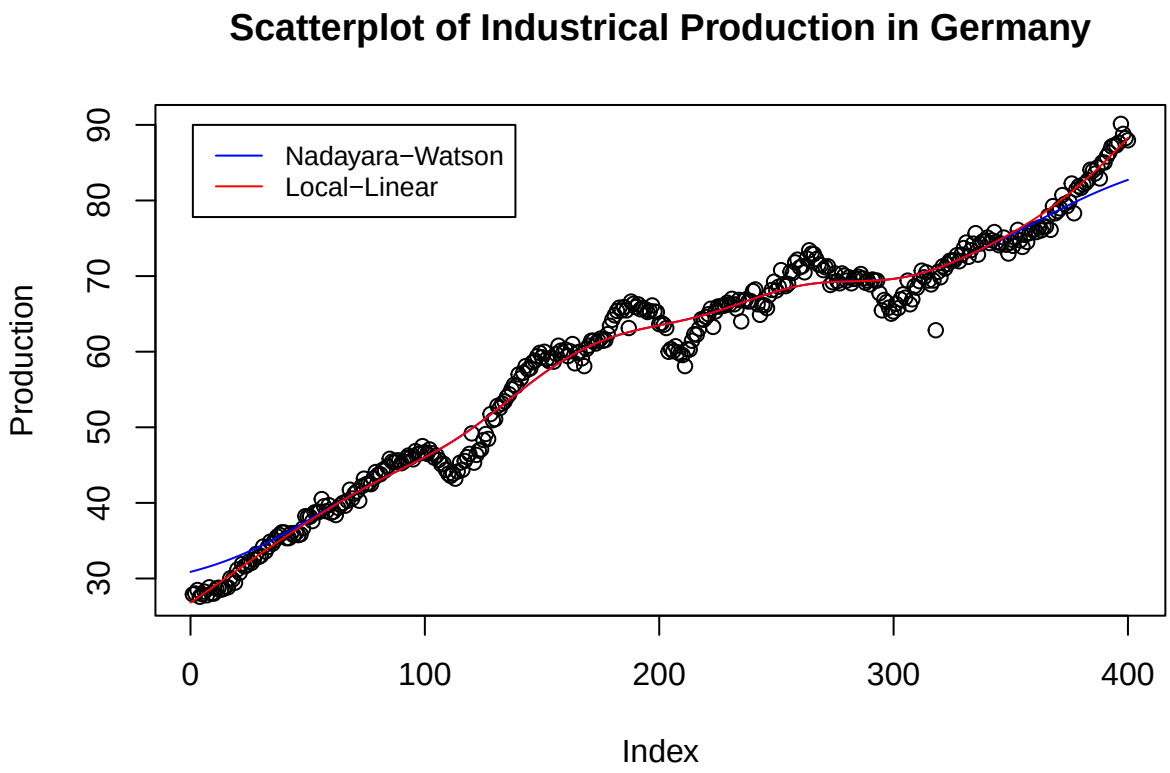


Part b
At both boundaries the curve is inaccurate, in other words, the curve is far from the data.
At the left boundary, the curve is overestimating the data, and on the right boundary the curve is underestimating the data.

Part c

```
# local-linear estimate and N-W estimate
reg.estimate.ll <- locpoly(x = x, y = germ$production,
                           degree = 1, bandwidth = 24)
plot(germ$production, main = "Scatterplot of Industrical Production in Germany", xlab = "Index", ylab =
```

```
lines(reg.estimate.NW, col = "blue")
lines(reg.estimate.ll, col = "red")
legend(1, 90, legend = c("Nadayara-Watson", "Local-Linear"), col = c("blue", "red"), lty = 1, cex = 0.8)
```



Part d
The local linear model with the same bandwidth as the Nadaraya-Watson model does adjust for the boundary problem. Now using the local linear estimate, we are no longer under or over estimating the points at either end of the data.

Problem 11

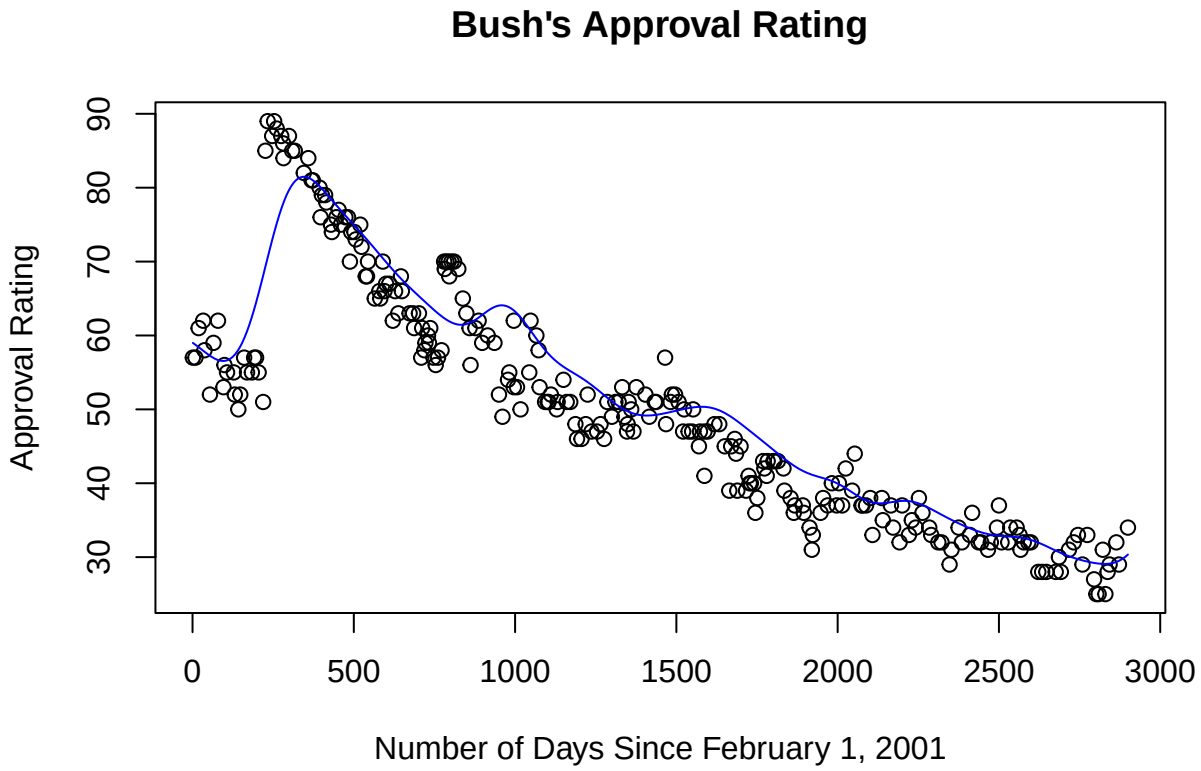
```
# Reading in the data
rating <- read_csv("WBushApproval.csv")
```

```
## Rows: 270 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): Start, Stop
## dbl (3): Approve, Disapprove, None
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
rating$Date <- as.Date(rating$Start, format = "%m/%d/%Y")

# Changing Date to number of days after February 1, 2001
rating <- rating %>% mutate(Date = as.numeric(rating$Date) - 11353)
```

Part a

```
# Local Regression Estimate
x3 <- seq(1, 2900, length = 270)
reg.est.bush.ll <- locpoly(x = x3 , y = rating$Approve,
                           degree = 1, bandwidth = 70)
plot(x = rating$Date, y = rating$Approve, main = "Bush's Approval Rating", xlab = "Number of Days Since
lines(reg.est.bush.ll, col = "blue")
```



Part b

```
# Calculating the number of days since February 1, 2001, to January 15, 2005
date_1 <- as.Date("2001-02-01")
date_2 <- as.Date("2005-06-15")
a = seq(from = date_1, to = date_2, by = 'day')
jun_15 <- length(a) - 1

# estimate for June 15, 2005
approx(reg.est.bush.ll$x, reg.est.bush.ll$y, xout = jun_15)
```

```
## $x
## [1] 1595
##
## $y
## [1] 50.3069
```

The estimated approval rating from June 15, 2005 using the local linear regression is 50.3069

Part c

```
# Fitting the Nadayara-Watson Estimate
reg.est.bush.NW <- locpoly(x = x3 , y = rating$Approve,
                           degree = 0, bandwidth = 70)

# need to compare both estimates on October 1, 2001
date_3 <- as.Date("2001-10-01")
b = seq(from = date_1, to = date_3, by = 'day')
oct_1 <- length(b) - 1

# Need to find estimate for October 1, 2001
# Local Linear estimate
approx(reg.est.bush.ll$x, reg.est.bush.ll$y, xout = oct_1)
```

```
## $x
## [1] 242
##
## $y
## [1] 72.08877
```

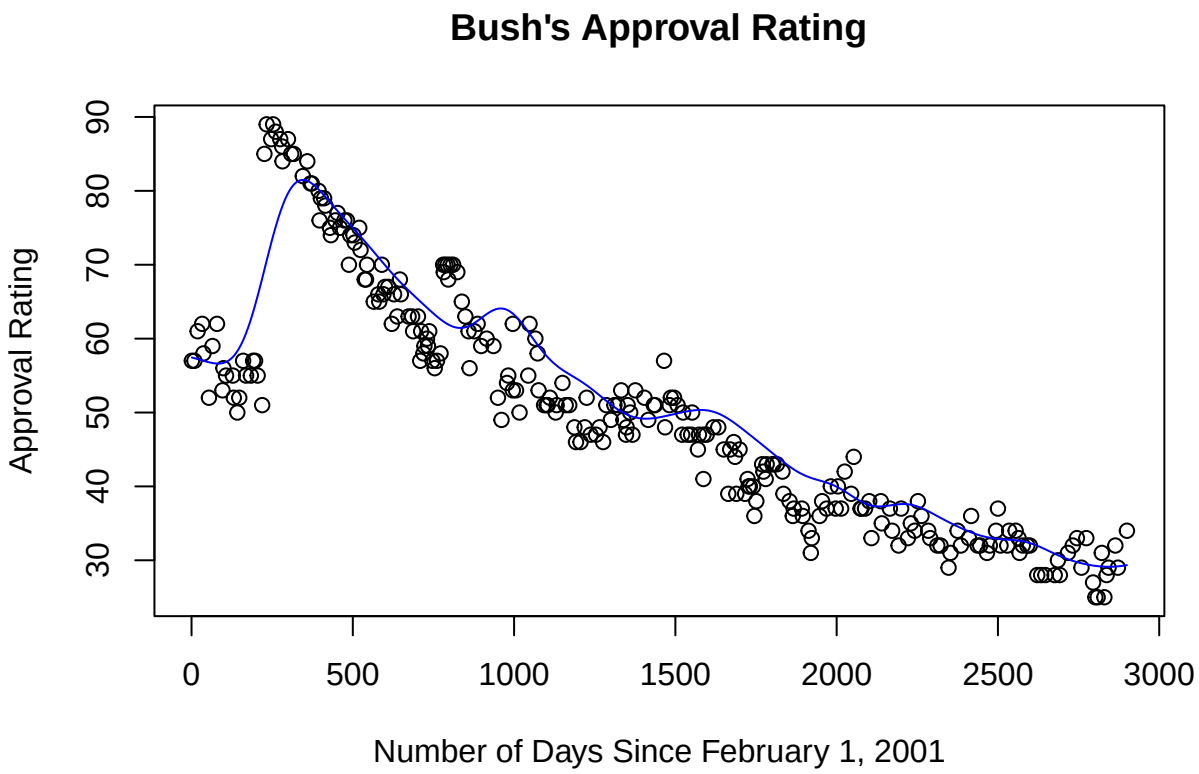
```
# Nadayara-Watson estimate
approx(reg.est.bush.NW$x, reg.est.bush.NW$y, xout = oct_1)
```

```
## $x
## [1] 242
##
## $y
## [1] 72.09711
```

The local linear estimate for the approval rating is 72.08877 while the Nadayara-Watson estimate is 71.13277. So both of the estimates are fairly close to each other.

Part d

```
# Plot of NW-Watson
plot(x = rating$Date, rating$Approve, main = "Bush's Approval Rating", xlab = "Number of Days Since Febr
lines(reg.est.bush.NW, col = "blue")
```



Part e
The discontinuity in this data at September 11, 2001 means that we cannot produce an accurate estimate for the discontinuous area. In other words it is difficult to connect the data from before September 11 to after September 11. One possible way to mitigate this effect is to produce two different models in order to fit the jump (ie discontinuity), one model for before September 11 and one model after. Another possibility would be to perform a log transformation on the data and from there get a more accurate and smooth curve.

Running Simulations to Produce Small Margins of Error

Problem 12

```
# Simulation
sim_data1 <- rexp(10^6, rate = 1)

# Calculating the probability of A
mean(cos(sim_data1) > 0.7)
```

```
## [1] 0.551541
```

```
# 95% Confidence Interval
p_hat <- mean(cos(sim_data1) > 0.7)
se1 <- sqrt((p_hat*(1-p_hat)/10^6))
c(p_hat - 1.96*se1, p_hat + 1.96*se1)
```

```
## [1] 0.5505662 0.5525158
```

From a simulation of one million random trials, the probability of event A is 0.551674
My 95% confidence interval for the probability of event A is (0.5506992, 0.5653979)

Problem 13

Part a

```
# Simulation
sim_data2 <- rbeta(378, 6, 1.5)

# IQR
IQR <- IQR(sim_data2)*100
IQR
```

```
## [1] 19.15057
```

The inter-quartile range of the simulation with 378 observations is 18.00144

Part b

```
# 10000 trials
ranges <- rep(NA, 10^4)
for (i in 1:10000){
  ranges[i] <- IQR(100*rbeta(378, 6, 1.5))
}

# estimated p-value
phat2 <- mean(ranges <= 17)
phat2
```

```
## [1] 0.0686
```

My estimated p-value is 0.0686

Part c

```
# Additional work in handwritten portion at the end of the pdf
N <- (0.0686*(1-0.0686))/((.0001/1.96)^2)
N
```

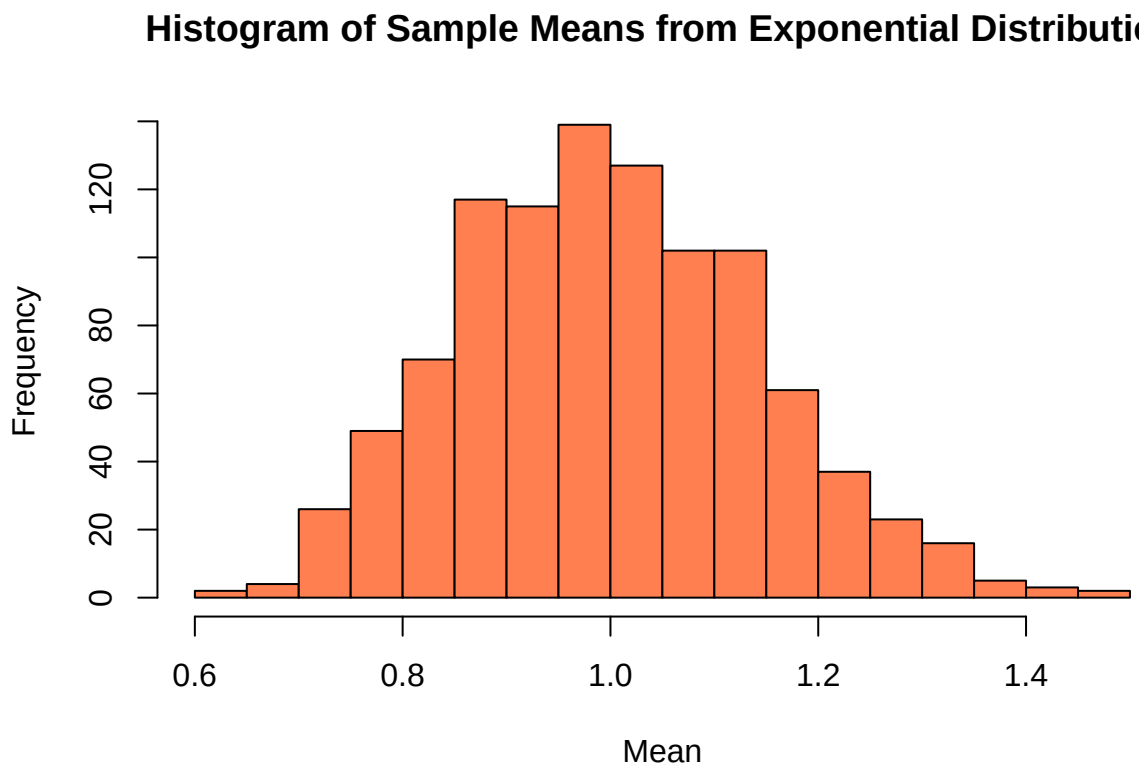
```
## [1] 24545534
```

In order for our margin of error to be less than 0.0001, I would have to run at least 24545535 simulations.

Problem 14

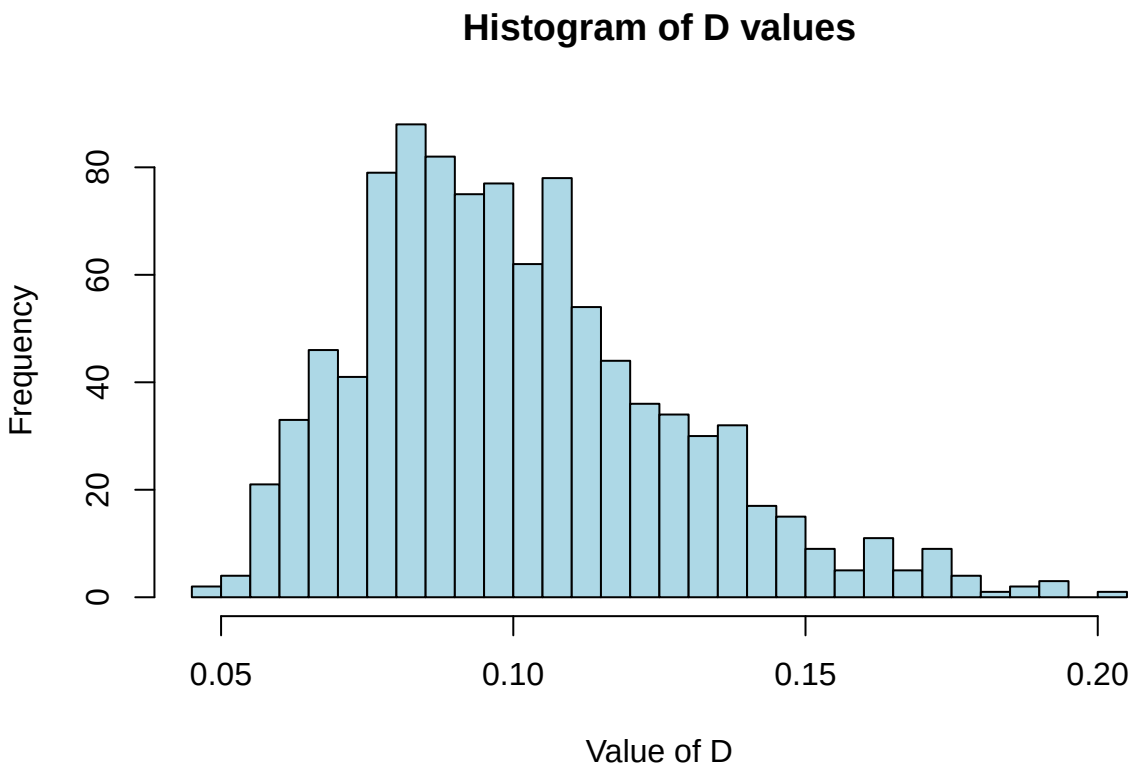
Part a

```r
# Creating a matrix with a sample in each column
sample_dat <- matrix(rexp(50*1000, rate = 1), nrow = 50)

# Mean estimate of each sample
sample_means <- colMeans(sample_dat)

# Histogram
hist(sample_means, breaks = 25, col = "coral", main = "Histogram of Sample Means from Exponential Distri
```



Part b

```r
# Performing KS test for each sample
D_values <- rep(NA, 1000)
for (i in 1:1000){
  ks <- ks.test(sample_dat[,i], "pexp", 1/mean(sample_dat[,i]))
  D_values[i] <- ks$statistic
}

# Histogram of D values
hist(D_values, col = "lightblue", breaks = 30, main = "Histogram of D values", xlab = "Value of D")
```



Part c

```
# Finding critical value
crit_val <- quantile(D_values, 0.95)
crit_val
```

```
##       95%
## 0.1488223
```

The critical value I obtained from the D values is 0.1517626

Part d
The following code is a simulation which takes two hours to run, hence it is just a chunk of text.

```
start <- Sys.time()
N <- 134996 # Number of batches
crit_vals <- rep(NA, N)
for (i in 1:N){
  D_values2 <- rep(NA, 1000)
  sample_dat2 <- matrix(rexp(50*1000, rate = 1), nrow = 50)
  for (j in 1:1000){
   ks2 <- ks.test(sample_dat2[,j], "pexp", 1/mean(sample_dat2[,j]))
   D_values2[j] <- ks2$statistic
  }
  crit_vals[i] <- quantile(D_values2, 0.95)
}
```

```
ending <- Sys.time()
total_time <- ending-start

# Calculating number of batches needed for 2 hours
# 100 batches takes 5.333523 seconds
# 7200 seconds in two hours
7200/5.333523 # 1349.952

# So I need
100*1349.952 # batches
# 134995.2 batches


total_time
# New critical value
new_crit <- quantile(crit_vals, 0.95)
new_crit

# Output

# Time difference of 1.960287 hours

#  95%
# 0.155033
```

My new critical value estimate (c) from my new simulations is 0.155033

Part e

```
D <- 1.094/(sqrt(50) + 0.26 + (0.5/sqrt(50))) + 0.2/50
D
```

```
## [1] 0.1518023
```

The critical value from Table 1.4 of Stephens(1974) paper suggests a value of 0.1518023. Considering I ran over 10^5 batches and got a critical value of 0.155033, I do think the value from Table 1.4 is significantly different. Although to the eye the difference seems very small, but the professor tested an estimate in class with a value from very many simulations, and the result was an extremely large z-score value. Which is why I am led to believe that there is a significant difference from my simulation results.