STAA 567: HW 2

Matthew Stoebe

Q1

Q1A

Write the Log likelyhood Function

Q1B

Find the First Derrivative

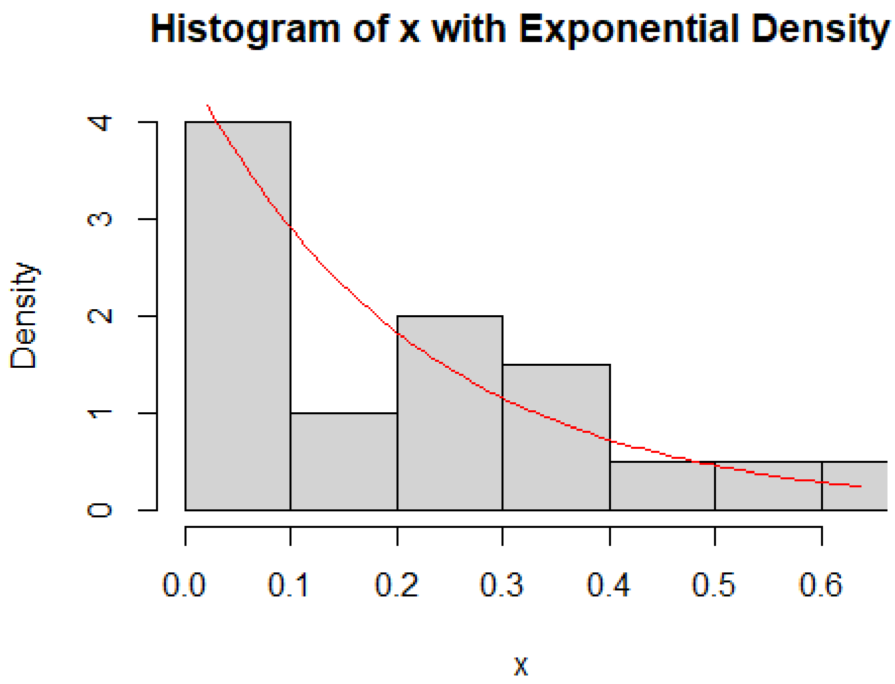
Q1C

Find the Fomula for Lambda hat

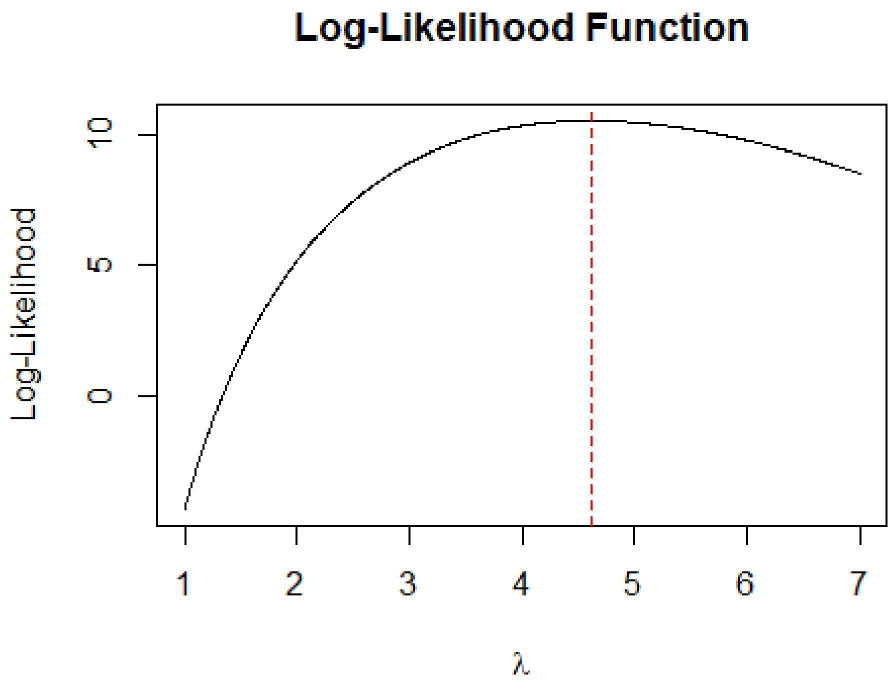
Q1D

```
## [1] 4.607375
```

Q1E


Histogram of x with Exponential Density
Density
0
1
2
3
4
0.0
0.1
0.2
0.3
0.4
0.5
0.6
x

Q1f


Log-Likelihood Function
Log-Likelihood
0
5
10
1
2
3
4
5
6
7
λ

```
## [1] 4.607375
```

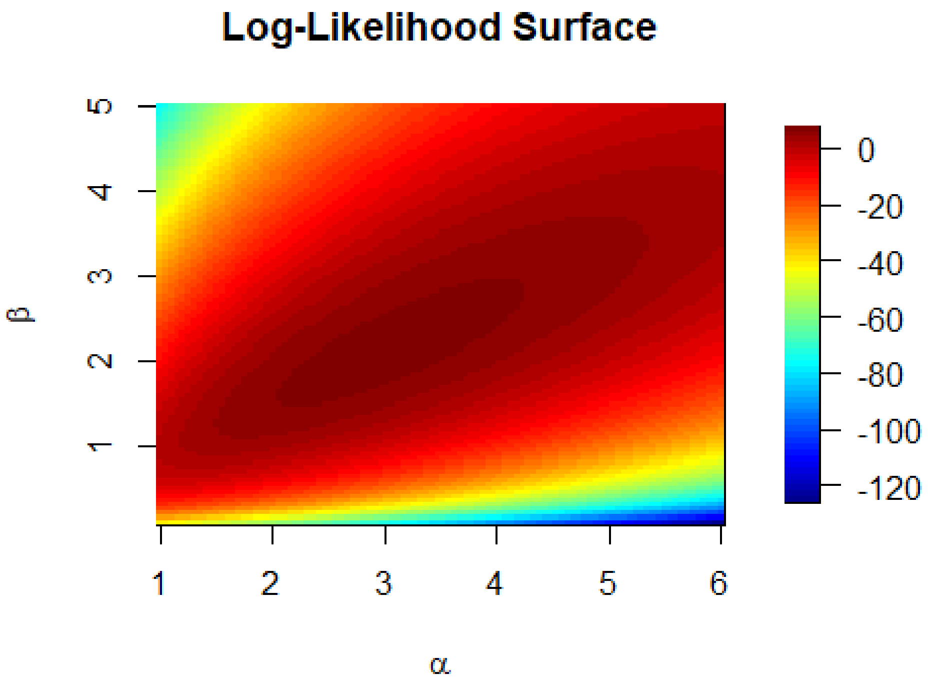
Q1I

```
## Iteration 1 : lambda = 4.04661
## Iteration 2 : lambda = 4.539124
## Iteration 3 : lambda = 4.606364
## Iteration 4 : lambda = 4.607375
## Iteration 5 : lambda = 4.607375

## [1] 4.607375
```

Q2

Q2A

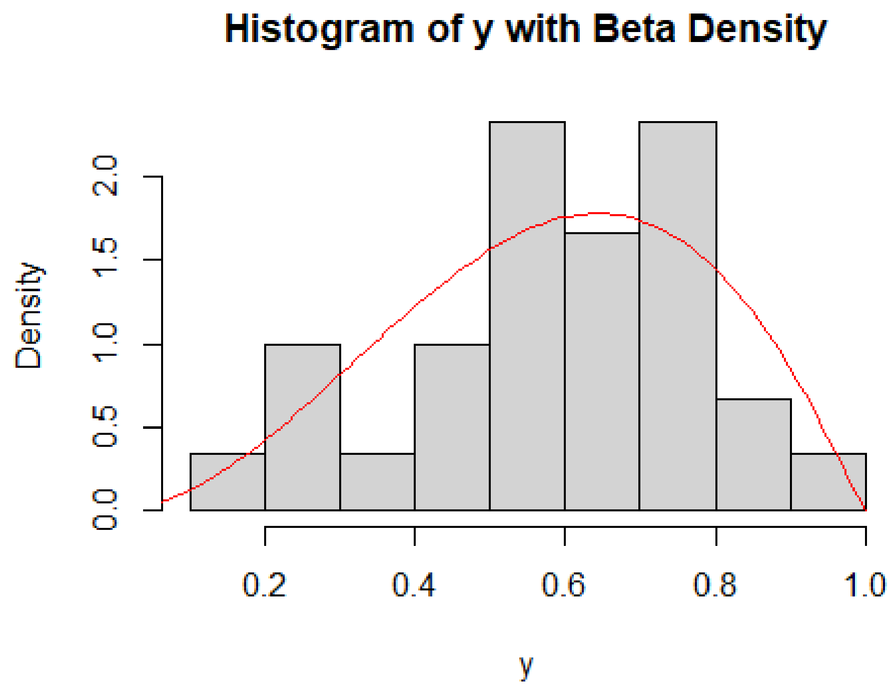


Q2B

```
##    alpha     beta 
## 3.014761 2.115292 

## function gradient 
##       55       NA 
```

Q2C



Q2D

```
##    alpha     beta
## 3.015065 2.115614

## function gradient
##       23        9
```

Appendix

```
#Retain this code chunk!!!
library(knitr)
knitr::opts_chunk$set(echo = FALSE)
knitr::opts_chunk$set(message = FALSE)
knitr::opts_chunk$set(warning = FALSE)

#install.packages("field")

#install.packages("tinytex")

#Q1D
load("./Data/expData.RData")

lambda_hat <- 1 / mean(x)
lambda_hat
```

```r
hist(x, freq = FALSE, xlim = c(0, max(x)), main = "Histogram of x with Exponential Density", xlab = "x")

x_vals <- seq(0, max(x), length.out = 100)
dens_vals <- dexp(x_vals, rate = lambda_hat)
lines(x_vals, dens_vals, col = "red")
lambda_seq <- seq(1, 7, length.out = 1000)

logLik_vals <- sapply(lambda_seq, function(lambda) {
  n <- length(x)
  n * log(lambda) - lambda * sum(x)
})

plot(lambda_seq, logLik_vals, type = 'l', xlab = expression(lambda), ylab = "Log-Likelihood", main = "Log-Likelihood Function")

abline(v = lambda_hat, col = 'red', lty = 2)

neg_logLik <- function(lambda) {
  if (lambda <= 0) return(Inf)
  n <- length(x)
  - (n * log(lambda) - lambda * sum(x))
}

opt_result <- optimize(neg_logLik, interval = c(1, 7))
lambda_hat_opt <- opt_result$minimum
lambda_hat_opt
newton_raphson <- function(x, lambda_init = 3, tol = 1e-6, max_iter = 100) {
  n <- length(x)
  S <- sum(x)
  lambda_old <- lambda_init
  for (i in 1:max_iter) {
    dL <- n / lambda_old - S
    d2L <- - n / lambda_old^2
    lambda_new <- lambda_old - dL / d2L
    cat("Iteration", i, ": lambda =", lambda_new, "\n")
    if (abs(lambda_new - lambda_old) < tol) {
      break
    }
    lambda_old <- lambda_new
  }
  return(lambda_new)
}

lambda_hat_nr <- newton_raphson(x)
lambda_hat_nr
#Q2A
load("./Data/betaData.RData")
```

```r
logLik_beta <- function(alpha, beta, y) {
  if (alpha <= 0 || beta <= 0) {
    return(NA)
  }
  sum(dbeta(y, shape1 = alpha, shape2 = beta, log = TRUE))
}

alpha_seq <- seq(1, 6, length.out = 100)
beta_seq <- seq(0.1, 5, length.out = 100)

logLik_vals <- matrix(NA, nrow = length(alpha_seq), ncol = length(beta_seq))

for (i in 1:length(alpha_seq)) {
  for (j in 1:length(beta_seq)) {
    logLik_vals[i, j] <- logLik_beta(alpha_seq[i], beta_seq[j], y)
  }
}

library(fields)
image.plot(alpha_seq, beta_seq, logLik_vals, xlab = expression(alpha), ylab =
expression(beta), main = "Log-Likelihood Surface")
#Q2B
neg_logLik_beta <- function(params, y) {
  alpha <- params[1]
  beta <- params[2]
  if (alpha <= 0 || beta <= 0) {
    return(Inf)
  }
  -sum(dbeta(y, shape1 = alpha, shape2 = beta, log = TRUE))
}

start_params <- c(alpha = 3, beta = 3)

result_nelder <- optim(start_params, neg_logLik_beta, y = y, method =
"Nelder-Mead")
result_nelder$par  # Estimated parameters
result_nelder$counts  # Number of iterations

#Q2C
hist(y, freq = FALSE, main = "Histogram of y with Beta Density", xlab = "y")

y_vals <- seq(0, 1, length.out = 100)
dens_vals <- dbeta(y_vals, shape1 = result_nelder$par[1], shape2 =
result_nelder$par[2])
lines(y_vals, dens_vals, col = 'red')
#Q2D
result_bfgs <- optim(start_params, neg_logLik_beta, y = y, method = "BFGS")
result_bfgs$par  # Estimated parameters
result_bfgs$counts  # Number of iterations
```