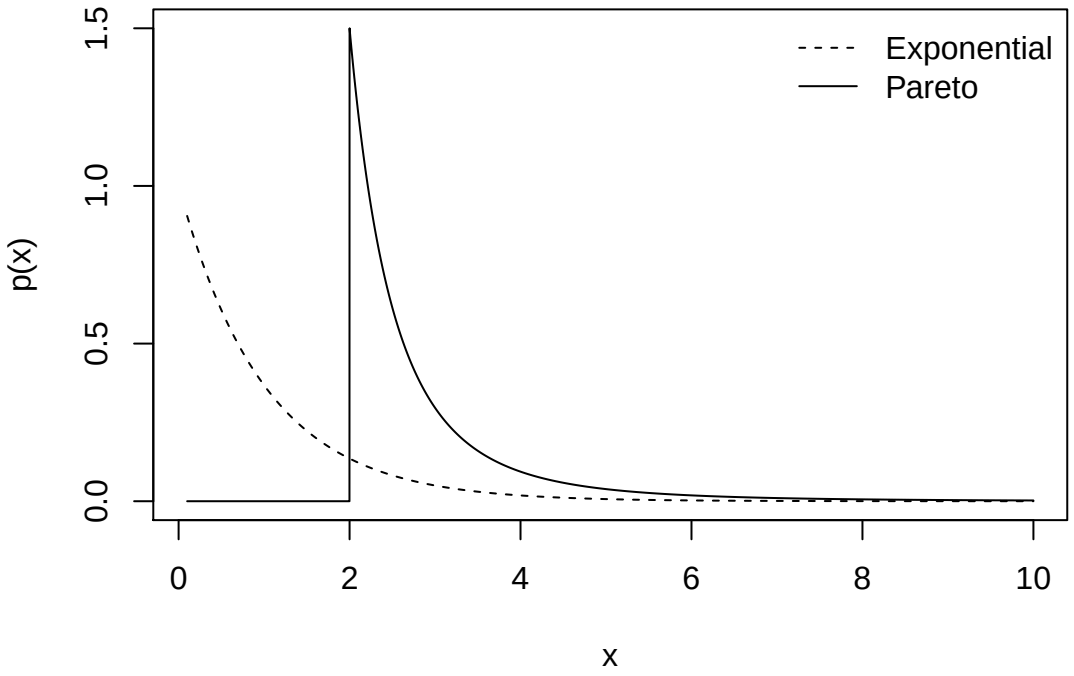
STAT243: Problem Set 8

Zicheng Huang

12/01/2017

1.(a)

```
library(VGAM)

## Loading required package:  stats4
## Loading required package:  splines

x <- seq(0.1, 10, length = 10000)
pareto <- dpareto(x, 2, 3)
plot(x, pareto, type = "l", ylab = "p(x)")
lines(x, dexp(x), lty = 2)
legend("topright", lty = c(2, 1), legend = c("Exponential", "Pareto"), bty = "n")
```



The tail of the Pareto decay more slowly than that of an exponential distribution.

1.(b)

First we estiamte $E(X)$ and $E(X^2)$

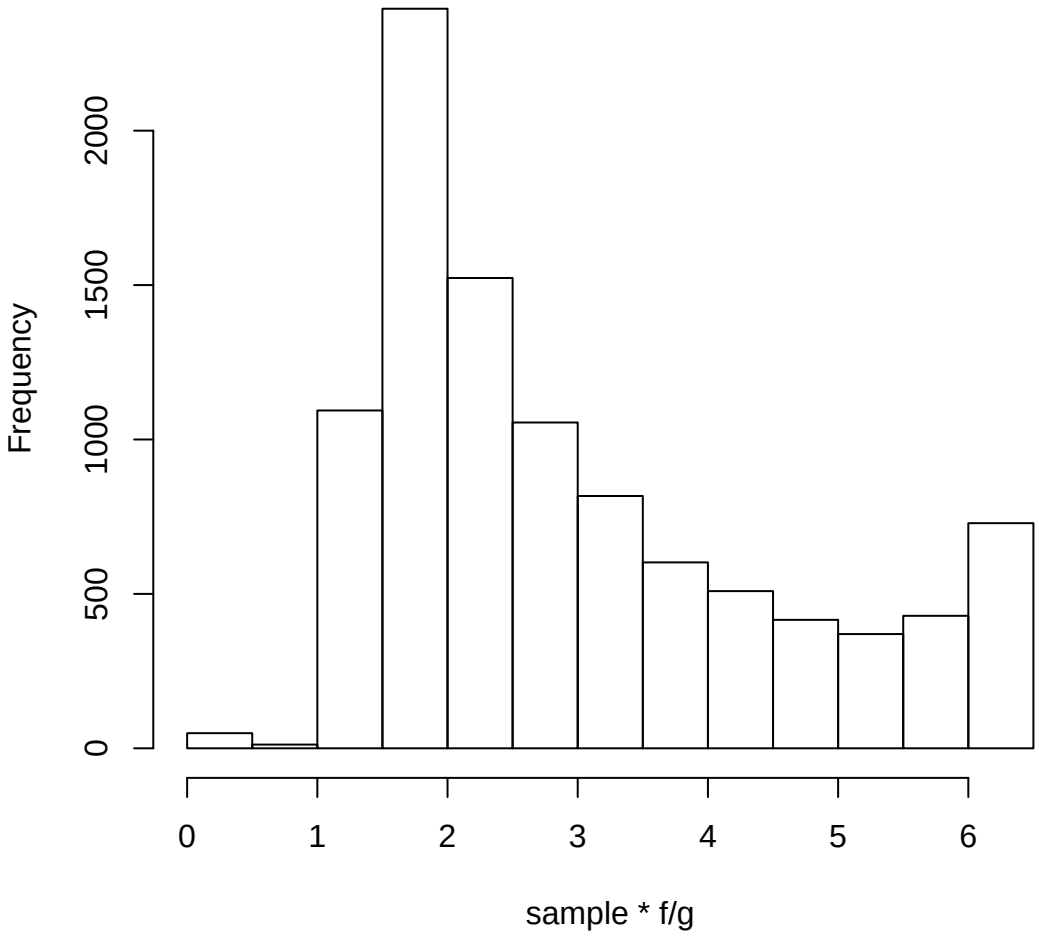
```
# setup
library(tolerance)
m <- 10000
alpha <- 2
beta <- 3

# get a sample of size 10000
sample <- rpareto(m, alpha, beta)

# f follows exponential density with parameter equals to 1, shifted by two to the right
f <- d2exp(sample, rate = 1, shift = 2)

# g follows Pareto distribution with alpha=2 and beta=3
g <- dpareto(sample, alpha, beta)

# estimate E(X)
mean(sample * f / g)
```

```
## [1] 3.003069
```

```
# calculate E(X^2)
mean(sample^2 * f / g)
```

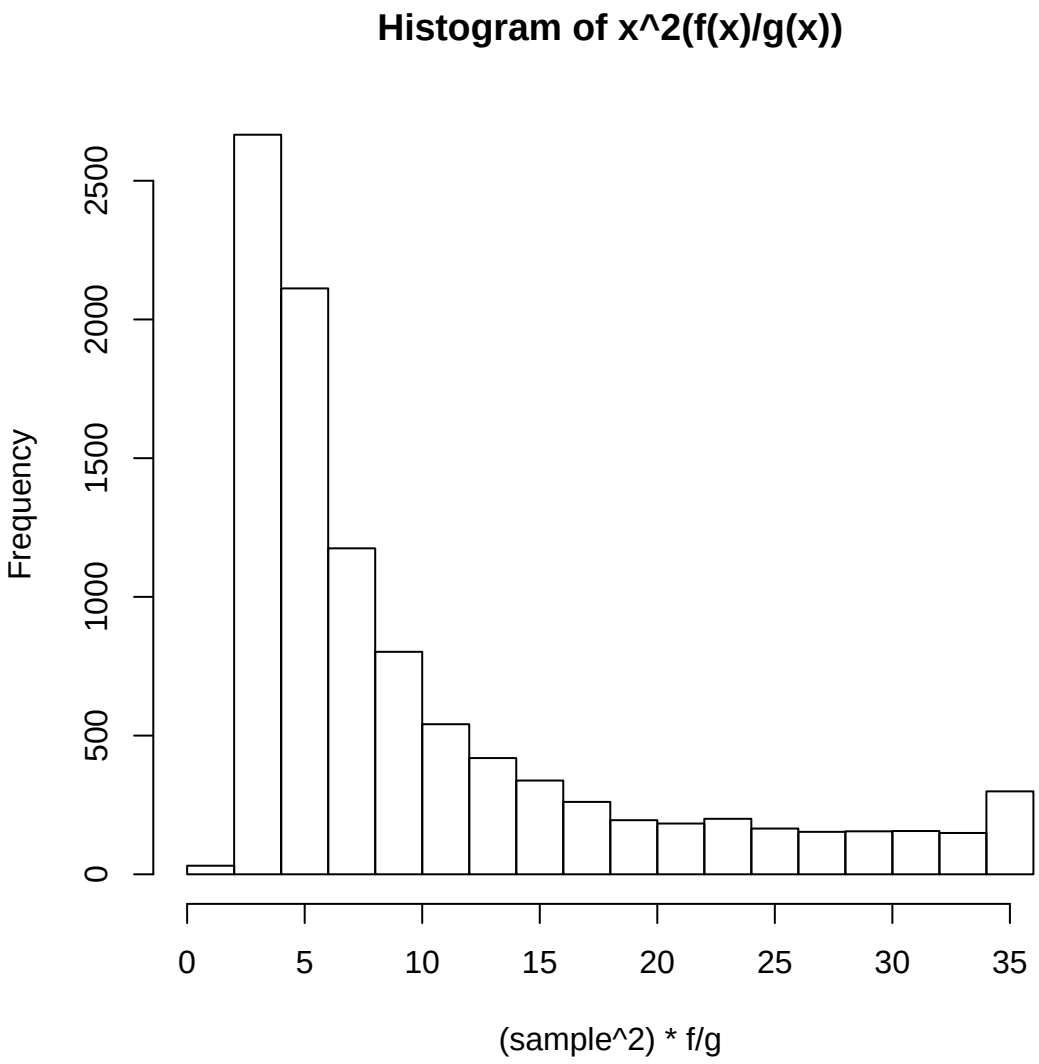
```
## [1] 10.03454
```

```r
# Histograms of h(x)f(x)/g(x), h(x) = x
hist(sample * f / g, main = "Histogram of x(f(x)/g(x))")
```



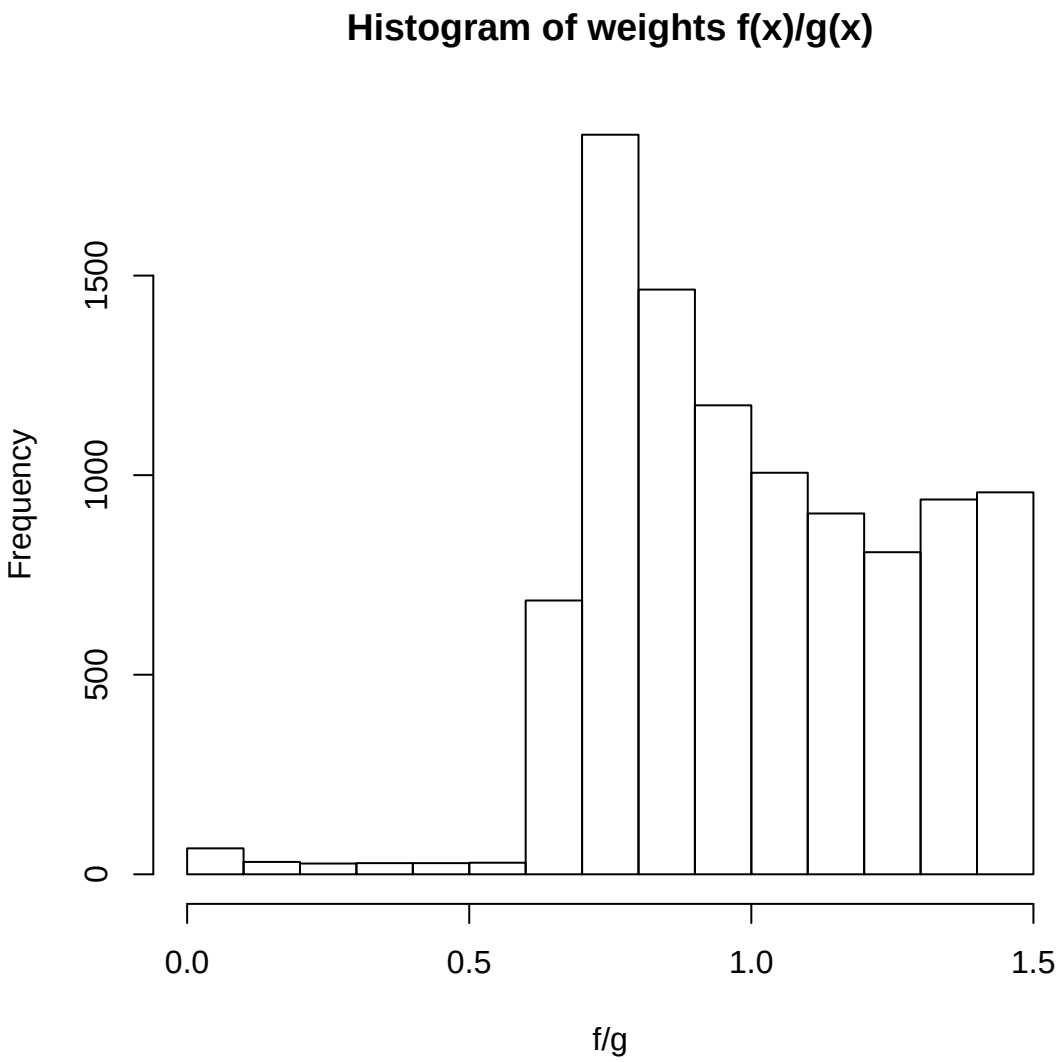

```r
# variance of x(f(x)/g(x))
var(sample * f / g)

## [1] 2.383916
```

```r
# Histograms of h(x)f(x)/g(x), h(x) = x^2
hist((sample^2) * f / g, main = "Histogram of x^2(f(x)/g(x))")
```




```r
# variance of x^2(f(x)/g(x))
var((sample^2) * f / g)

## [1] 76.26334
```

```r
# Histogram of weights f(x)/g(x)
hist(f / g, main = "Histogram of weights f(x)/g(x)")
```



For importance sampling, we would like to have a sampling distribution that decays more slowly than the distribution of interest, in other words, having a heavier tail. This is the case we have here as the sampling distibution follows a Pareto distibution which has heavier tail than that of exponential distribution. Based on the shape of the histograme, together with the computed variance $Var(\hat{\phi})$, we can see that the variance is relatively small. Thus, this indicates that there is no extreme weights that would have a very strong influence on $\hat{\mu}$, which can be observed from the narrow range of weights in the histogram above.

1.(c)

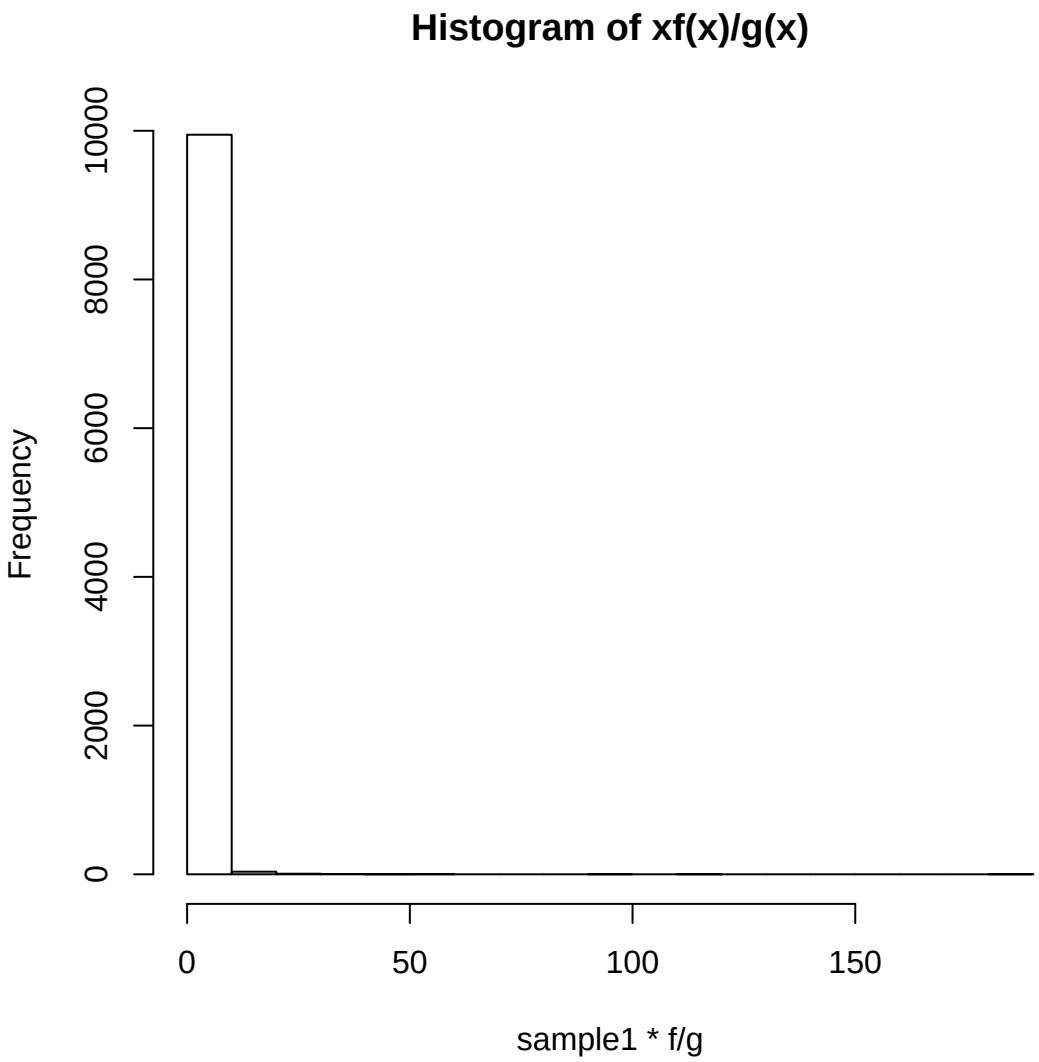
```r
# setup
m <- 10000
alpha <- 2
beta <- 3

# get a sample of size 10000 from the shifted exponential distribution
sample1 <- r2exp(m, rate = 1, shift = 2)

# f follows Pareto distribution with alpha=2 and beta=3
f <- dpareto(sample1, alpha, beta)

# g follows exponential density with parameter equals to 1, shifted by two to the right
g <- d2exp(sample1, rate = 1, shift = 2)

# estimate E(X)
mean(sample1 * f / g)
```

```
## [1] 2.899689
```

```r
# estimate E(X^2)
mean((sample1^2) * f / g)
```

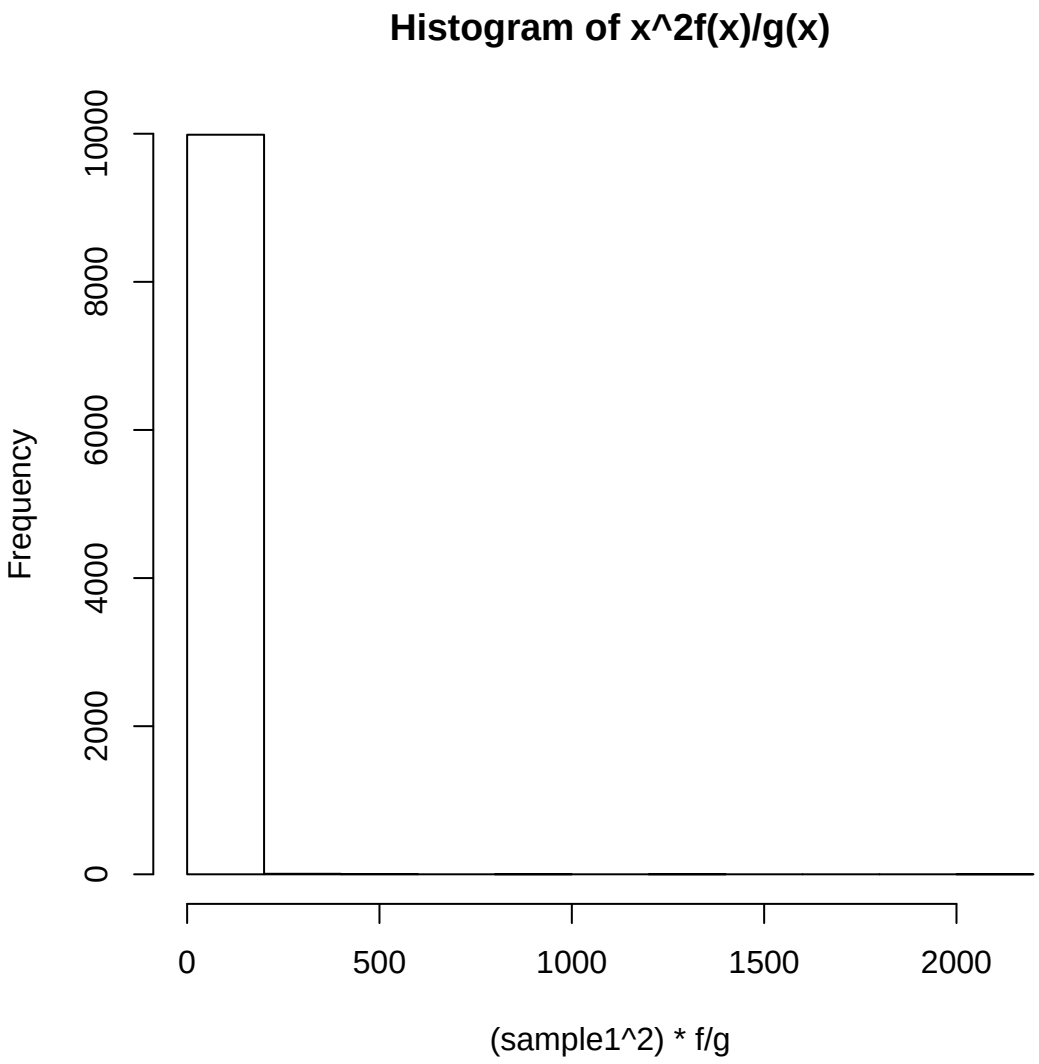
```
## [1] 9.837696
```

```r
# Histograms of h(x)f(x)/g(x), h(x) = x
hist(sample1 * f / g, main = "Histogram of xf(x)/g(x)")
```



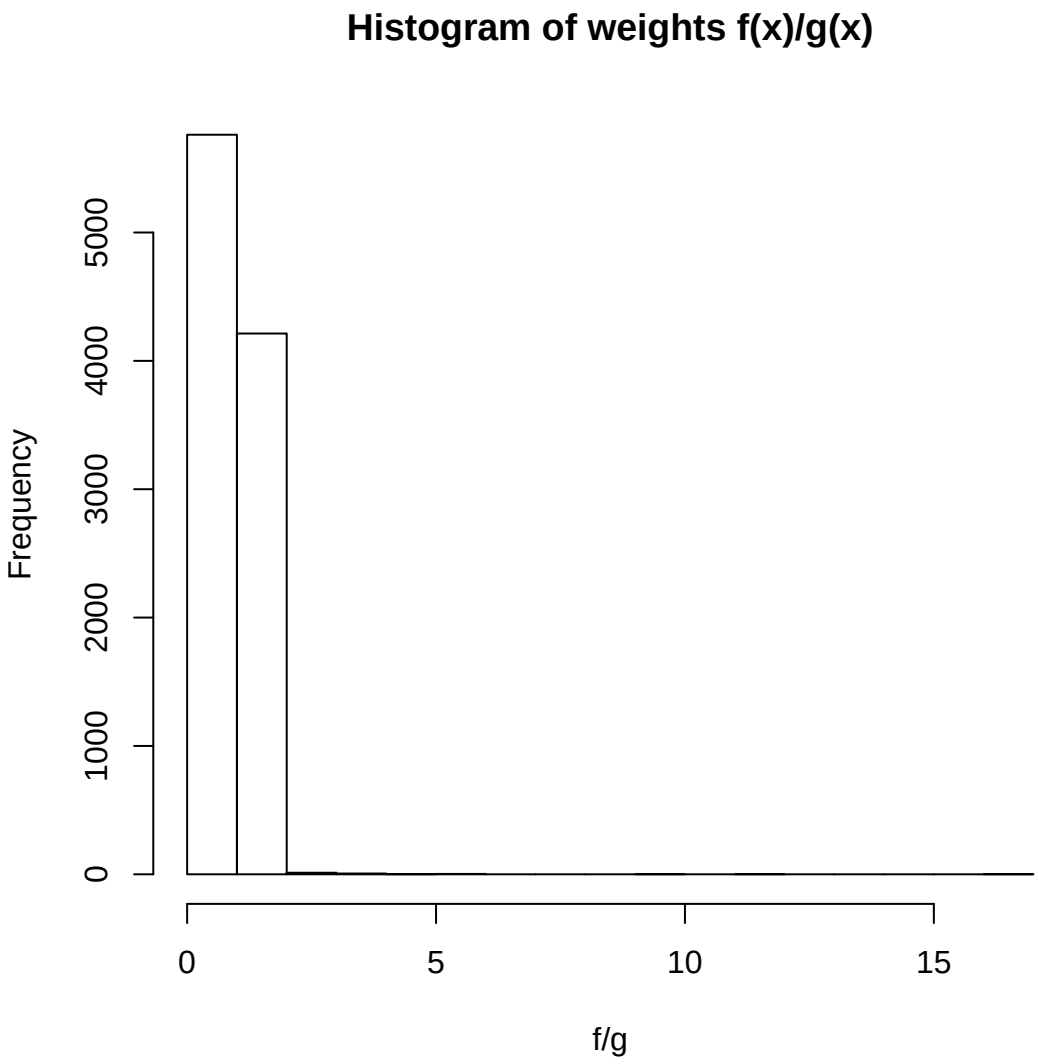

```r
# variance of x(f(x)/g(x))
var(sample1 * f / g)

## [1] 8.101532
```

```r
# Histograms of h(x)f(x)/g(x), h(x) = x^2
hist((sample1^2) * f / g, main = "Histogram of x^2f(x)/g(x)")
```




```r
# variance of x^2(f(x)/g(x))
var((sample^2) * f / g)

## [1] 60714.01
```

```r
# Histogram of weights f(x)/g(x)
hist(f / g, main="Histogram of weights f(x)/g(x)")
```



In this case, we are sample from a distribution decaying faster than the density of interest, thus should have a larger variance $Var(\hat{\phi})$. This can be observed from the first two histograms. Moreover, the computed variance under each histogram also shows that the variance is relatively larger. This indicates that there should be some extreme weights that would have a very strong influence on $\hat{\mu}$, which is also observable from the wider range of weights in the histogram above.

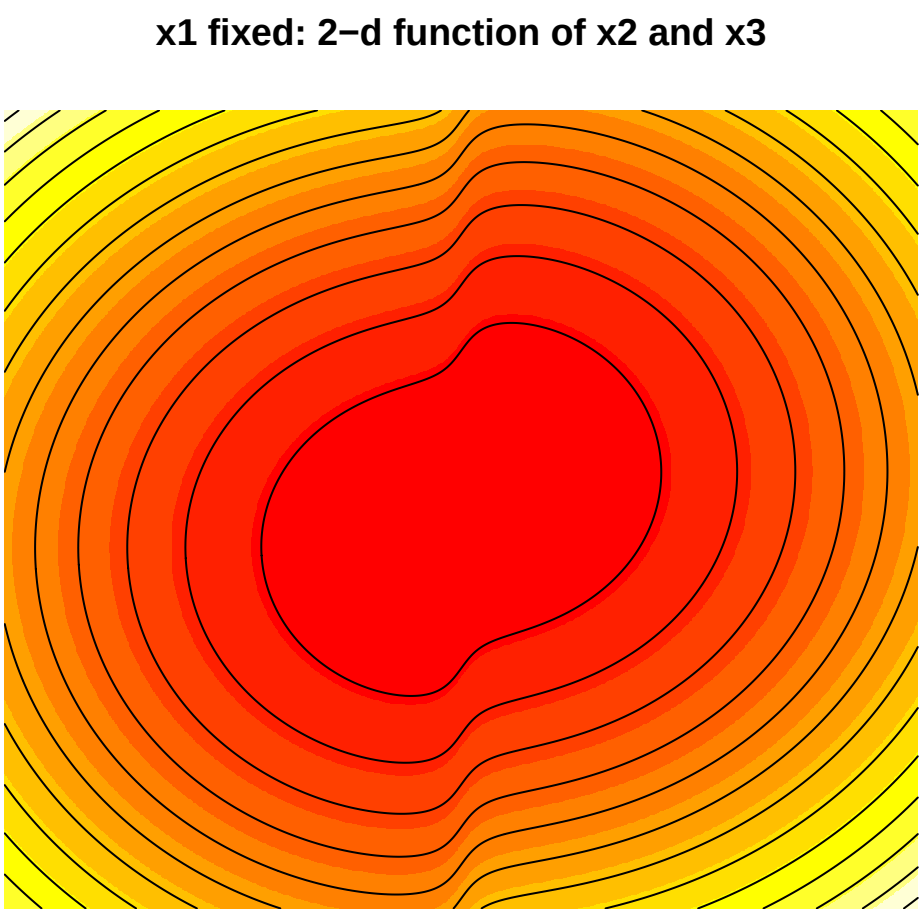
2

Setup:

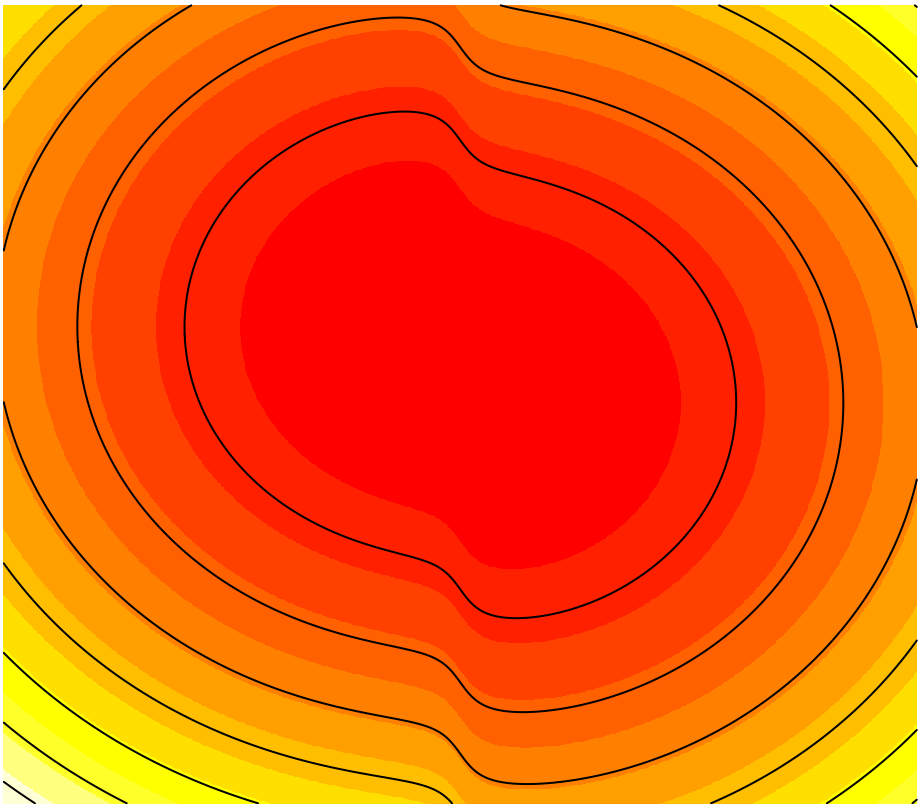
```
# "helical valley" function
theta <- function(x1,x2) atan2(x2, x1)/(2*pi)
f <- function(x) {
  f1 <- 10*(x[3] - 10*theta(x[1],x[2]))
  f2 <- 10*(sqrt(x[1]^2 + x[2]^2) - 1)
  f3 <- x[3]
  return(f1^2 + f2^2 + f3^2)
}

# define values for x1, x2, x3
x1 <- seq(-25,25, length = 1000)
x2 <- seq(-25,25, length = 1000)
x3 <- seq(-25,25, length = 1000)
```

Now we want to plot slices of the funciton to get a sense for how it behaves:

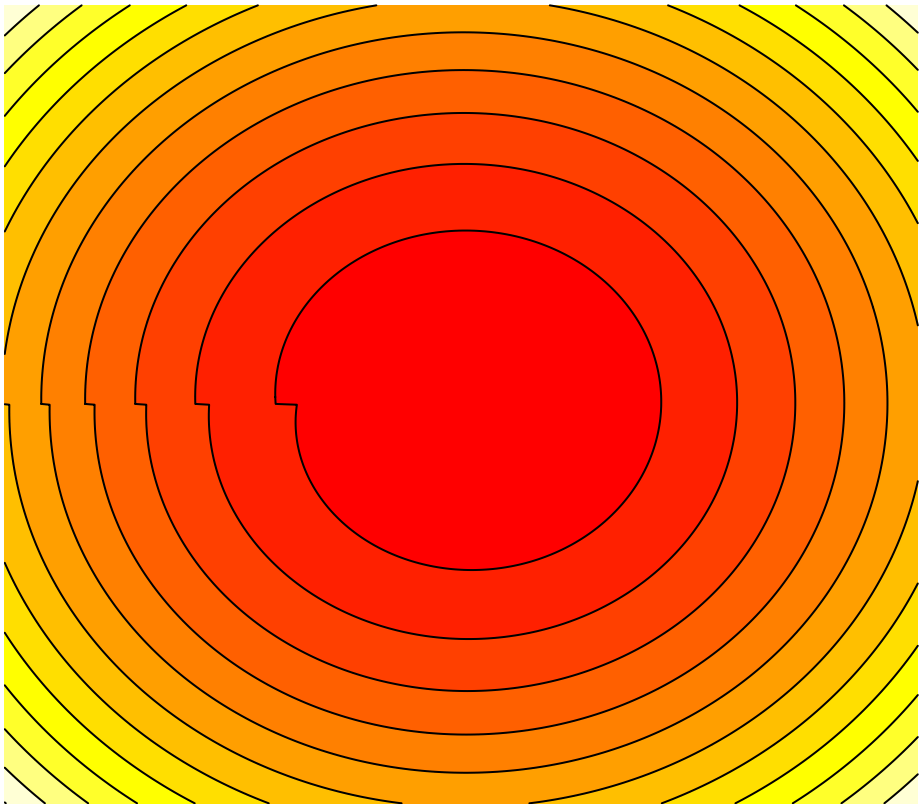
```
# x1 fixed; x2 and x3 vary
f23 <- outer(x2,              # this gives all possibilities of f(x2, x3)
             x3,              # while x1 is fixed at 1
             Vectorize(function(x2, x3){f(c(1, x2, x3))}))
# plot slice of f where x1 treated as constant value
image(f23, main = "x1 fixed: 2-d function of x2 and x3", axes = F)
contour(f23, add = T, drawlabels = F)
```



```r
# x2 fixed; x1 and x3 vary
f13 <- outer(x1,              # this gives all possibilities of f(x1, x3)
             x3,              # while x2 is fixed at 1
             Vectorize(function(x1, x3){f(c(x1, 1, x3))}))
# # plot slice of f where x2 treated as constant value
image(f13, main = "x2 fixed: 2-d function of x1 and x3", axes = F)
contour(f13, add = T, drawlabels = F)
```



```r
# x3 fixed; x1 and x2 vary
f12 <- outer(x1,              # this gives all possibilities of f(x1, x2)
             x2,              # while x3 is fixed at 1
             Vectorize(function(x1, x2){f(c(x1, x2, 1))}))
# # plot slice of f where x3 treated as constant value
image(f12, main = "x3 fixed: 2-d function of x1 and x2", axes = F)
contour(f12, add = T, drawlabels = F)
```



Now we want to explore the possibilities of multiple local minima by using different starting points:

```
# minimum value using optim() with starting point (1,1,1)
optim(c(1, 1, 1), f)$value

## [1] 1.343098e-07

# minimum value using nlm() with starting point (1,1,1)
nlm(f, c(1, 1, 1))$minimum

## [1] 1.702065e-08

# generate random starting values
x1 <- runif(10, -10, 10)
x2 <- runif(10, -10, 10)
x3 <- runif(10, -10, 10)

# Explore the possibility of multiple local minima with optim()
n <- 10
optimMin <- c()
for (i in 1:n) {
  result <- optim(c(x1[i], x2[i], x3[i]), f)
  optimMin <- c(optimMin, result$value)
}
# minimum values obtain using optim(), all observed to be different
optimMin

##  [1] 6.741497e-05 6.290605e-05 4.166127e-06 5.064648e-05 1.347556e-05
##  [6] 5.341045e-06 3.360065e-05 1.767904e-04 4.912437e-05 7.451741e-05

# Explore the possibility of multiple local minima with nlm()
n <- 10
nlmMin <- c()
for (i in 1:n) {
  result <- nlm(f, c(x1[i], x2[i], x3[i]))
  nlmMin <- c(nlmMin, result$minimum)
}
# minimum values obtain using nlm(), all observed to be different
nlmMin

##  [1] 8.027068e+01 6.329970e-15 1.247436e-15 6.237364e-18 1.700911e-08
##  [6] 1.700964e-08 8.214941e-16 2.124617e-20 1.172112e-16 1.003268e-15
```

As we can see, the minimum values correspond to different starting points are different. Therefore, this suggests that there is the possibility of multiple local minima.

3.(a)

Let $Y_{obs} = \{y_1, \cdots, y_{n-c}\}$ denotes the available observations of Y.

Let $m_t = \{m_{1,t}, m_{2,t}, \cdots, m_{c,t}\}$ be used in place of the censored observations.

Let $\mu_i = \beta_0 + \beta_1 x_i$

$$\begin{aligned}
\log L(\theta|Y) &= \log L(\theta|Y_{obs}, m_t) \\
&= \log L(\theta|y_1, \cdots, y_{n-c}, m_{1,t}, \cdots, m_{c,t}) \\
&= \log\Big[\prod_{i=1}^{n-c} \frac{1}{\sqrt{2\pi}\sigma} \exp\{-\frac{(y_i-\mu_i)^2}{2\sigma^2}\} \prod_{i=1}^{c} \frac{1}{\sqrt{2\pi}\sigma} exp\{-\frac{(m_{i,t}-\mu_i)^2}{2\sigma^2}\}\Big] \\
&= -\frac{n}{2}\log(2\pi) - n\log(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n-c}(y_i-\mu_i)^2 - \frac{1}{2\sigma^2}\sum_{i=1}^{c}(m_{i,t}-\mu_i)^2
\end{aligned}$$

Since $m_{i,t} \sim W|W > \tau$, we have:

$$\begin{aligned}
E[m_{i,t}] &= \mu_{i,t} + \sigma_t \rho(\tau^*_{i,t}) \\
Var[m_{i,t}] &= \sigma_t^2\big(1 + \tau^*_{i,t}\rho(\tau^*_{i,t}) - \rho(\tau^*_{i,t})^2\big) \\
\rho(\tau_t^*) &= \frac{\phi(\tau_t^*)}{1-\Phi(\tau_t^*)} \\
\tau_t^* &= \frac{\tau - \mu_{i,t}}{\sigma_t}
\end{aligned}$$

Therefore,

$$\mathbb{E}[m_{i,t} - \mu_{i,t}]^2 = \mathbb{E}[m_{i,t}^2] - 2\mathbb{E}[m_{i,t}]\mu_{i,t} + \mu_{i,t}^2 = (\mathbb{E}[m_{i,t}] - \mu_{i,t})^2 + Var(m_{i,t})$$

EM Algorithm:

E Step: compute $Q(\theta|\theta_t)$

$$\begin{aligned}
Q(\theta|\theta_t) &= \mathbb{E}(\log L(\theta|Y_{obs}, m_t)|x, \theta_t) \\
&= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n-c}(y_i-\mu_i)^2 - \frac{1}{2\sigma^2}\sum_{i=1}^{c}\big[(\mathbb{E}[m_{i,t}] - \mu_{i,t})^2 + Var(m_{i,t})\big]
\end{aligned}$$

M step: maximize $Q(\theta|\theta_t)$ with respect to θ to get θ_{t+1} by setting $\frac{d}{d\theta_t}Q(\theta|\theta_t) = 0$

$$\frac{d}{d\beta_1}Q(\theta|\theta_t) = 0 \implies \beta_{1,t+1} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\frac{d}{d\beta_0}Q(\theta|\theta_t) = 0 \implies \beta_{0,t+1} = \frac{1}{n}\Big(\sum_{i=1}^{n-c} y_i + \sum_{i=1}^{c}\mathbb{E}[m_{i,t}]\Big) + \frac{\sum_{i=1}^n x_i}{n}\beta_{1,t+1} = \bar{y} - \beta_{1,t+1}\bar{x}$$

$$\frac{d}{d\sigma^2}Q(\theta|\theta_t) = 0 \implies \sigma^2_{t+1} = \frac{1}{n}\Big(\sum_{i=1}^{n-c}(y_i - \beta_{0,t+1} - \beta_{1,t+1}x_i)^2 + \sum_{i=1}^{c} var(m_{i,t}) + \sum_{i=1}^{c}(\mathbb{E}[m_{i,t}] - \mu_{i,t})^2\Big)$$

3.(b)

We can choose a reasonable staring point by using the set of available observations in $Y_{obs} = \{y_1, \cdots, y_{n-c}\}$ with corresponding $X_{obs} = \{x_1, \cdots, x_{n-c}\}$ to perform a linear regression of Y_{obs} on X_{obs}. In this way, we can obtain a $\beta_{0,0}$ and $\beta_{1,0}$ where $\beta_{0,0} = \bar{y} - \beta_{1,0}\bar{x}$ and $\beta_{1,0} = \frac{\sum_{i=1}^{n-c}(x_i-\bar{x})(y_i-\bar{y})}{\sum_{i=1}^{n-c}(x_i-\bar{x})^2}$ and $\bar{x} = \frac{1}{n}\sum_1^{n-c} x_i$ and $\bar{y} = \frac{1}{n}\sum_1^{n-c} y_i$. The starting value σ_0^2 can be chosen as $\sigma_0^2 = \frac{\sum_{i=1}^{n-c}(y_i-\beta_{0,0}-\beta_{1,0}x_i)^2}{n-c-2}$ since this is a unbiased estimator. In this way, we have obtain the starting point $(\beta_{0,0}, \beta_{1,0}, \sigma_0^2)$.

3.(c)

```
#### c
rm(list = ls())

set.seed(1)
n <- 100
beta0 <- 1
beta1 <- 2
sigma2 <- 6

x <- runif(n)
yComplete <- rnorm(n, beta0 + beta1*x, sqrt(sigma2))

## parameters chose such that signal in data is moderately strong
## estimate divided by std error is ~ 3
mod <- lm(yComplete ~ x)
summary(mod)$coef
```

```
##              Estimate Std. Error  t value    Pr(>|t|)
## (Intercept) 0.5607442  0.5041346 1.112290 0.268734381
## x           2.7650812  0.8657927 3.193699 0.001889262
```

```
#get the criteria infomation about lm(yComplete ~ x)
beta0lm <- summary(lm(yComplete ~ x))$coef[1,1]
beta1lm <- summary(lm(yComplete ~ x))$coef[2,1]
sigmaSquarelm <- sum((yComplete - beta0lm - beta1lm * x)^2)/n
criteria <- c(beta0lm, beta1lm, sigmaSquarelm)


EM <- function(x, y, numPercent){

  # numPercent is the number of the percentage of the data that is censored
  tau <- sort(y, decreasing = TRUE)[numPercent]
  xUncensored <- x[y <= tau]
  yUncensored <- y[y <= tau]
  xCensored <- x[y > tau]
  yCensored <- y[y > tau]
  # n is total observations
  n <- length(y)
  # c is censored observations
  c <- length(yCensored)
```

```
  # initial starting values based on part(b)
  mod <- lm(yUncensored ~ xUncensored)
  beta0 <- summary(mod)$coef[1,1]
  beta1 <- summary(mod)$coef[2,1]
  sigmaSquare <- sum((yUncensored - beta0 - beta1 * xUncensored)^2)/(n - c -2)

  # iternation
  i <- 1
  # error between current value and previous value
  error <- 1
  while (error > 0.00001){

    # value used to calculate error
    beta00 <- beta0
    beta10 <- beta1
    sigmaSquare0 <- sigmaSquare

    # According to the the Johnson and Kotz bibles on distributions
    muCensored <- beta0 + beta1 * xCensored
    tauStar <- (tau - muCensored)/sqrt(sigmaSquare)
    rho <- dnorm(tauStar)/(1-pnorm(tauStar))
    mExpectation <- muCensored + sqrt(sigmaSquare)* rho
    mVariance <- sigmaSquare * (1+ tauStar*rho - rho^2)

    # rearrange to obtain xStar
    xStar <- c(xCensored, xUncensored)
    # add estimated value into yUncersored to obtain yStar
    yStar <- c(mExpectation, yUncensored)

    # obtain the estimations
    modStar <- lm(yStar ~ xStar)
    beta1 <- summary(modStar)$coef[2,1]
    beta0 <- summary(modStar)$coef[1,1]
    sigmaSquare <- (sum(summary(modStar)$residuals^2)  + sum(mVariance))/100

    # stop the optimization if the sum of the sqared value of "error" of the
    # three parameters is smaller than 0.00001
    error <- ((beta00-beta0)^2 + (beta10-beta1)^2 + (sigmaSquare0-sigmaSquare)^2)
    i = i + 1
  }
  return(list(Beta0 = beta0, Beta1 = beta1, SigmaSquare = sigmaSquare))
}
```

```r
# criteria information obtained from regression of yComplete on x
criteria
```

```
## [1] 0.5607442 2.7650812 5.2072910
```

```r
# 20 percent of the data is censored
EM(x, yComplete, 20)
```

```
## $Beta0
## [1] 0.4879929
##
## $Beta1
## [1] 2.738818
##
## $SigmaSquare
## [1] 4.562418
```

```r
# 80 percent of the data is censored
EM(x, yComplete, 80)
```

```
## $Beta0
## [1] 0.4029393
##
## $Beta1
## [1] 2.434516
##
## $SigmaSquare
## [1] 3.613725
```

Comparing to the criteria values, when 20 percent of the data is censored, the estimations of the coefficients and sigma square is better than the estimations when 80 percnet of the data is censored.