Problem Set 8

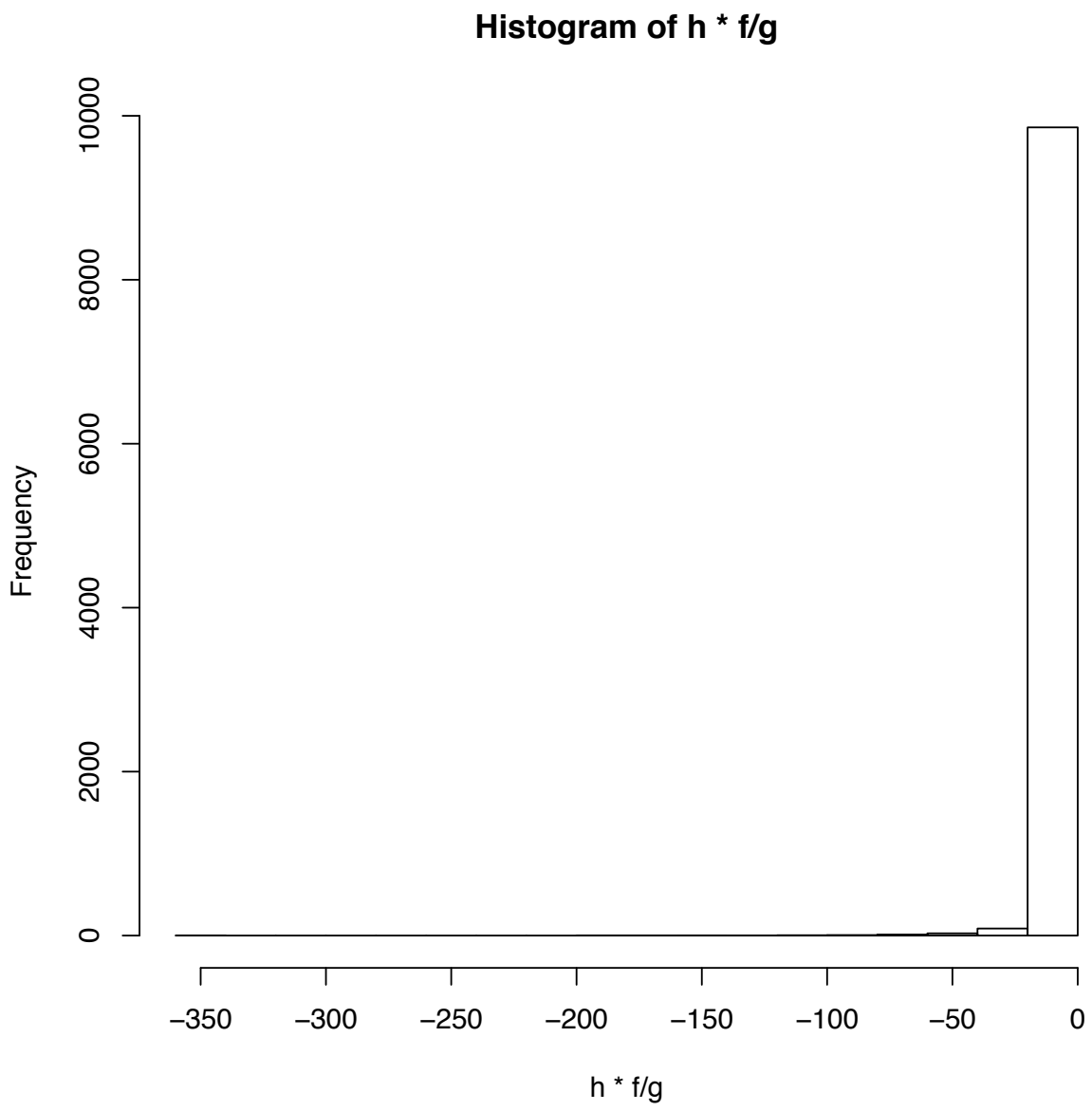
Stephanie Kim

December 3, 2014

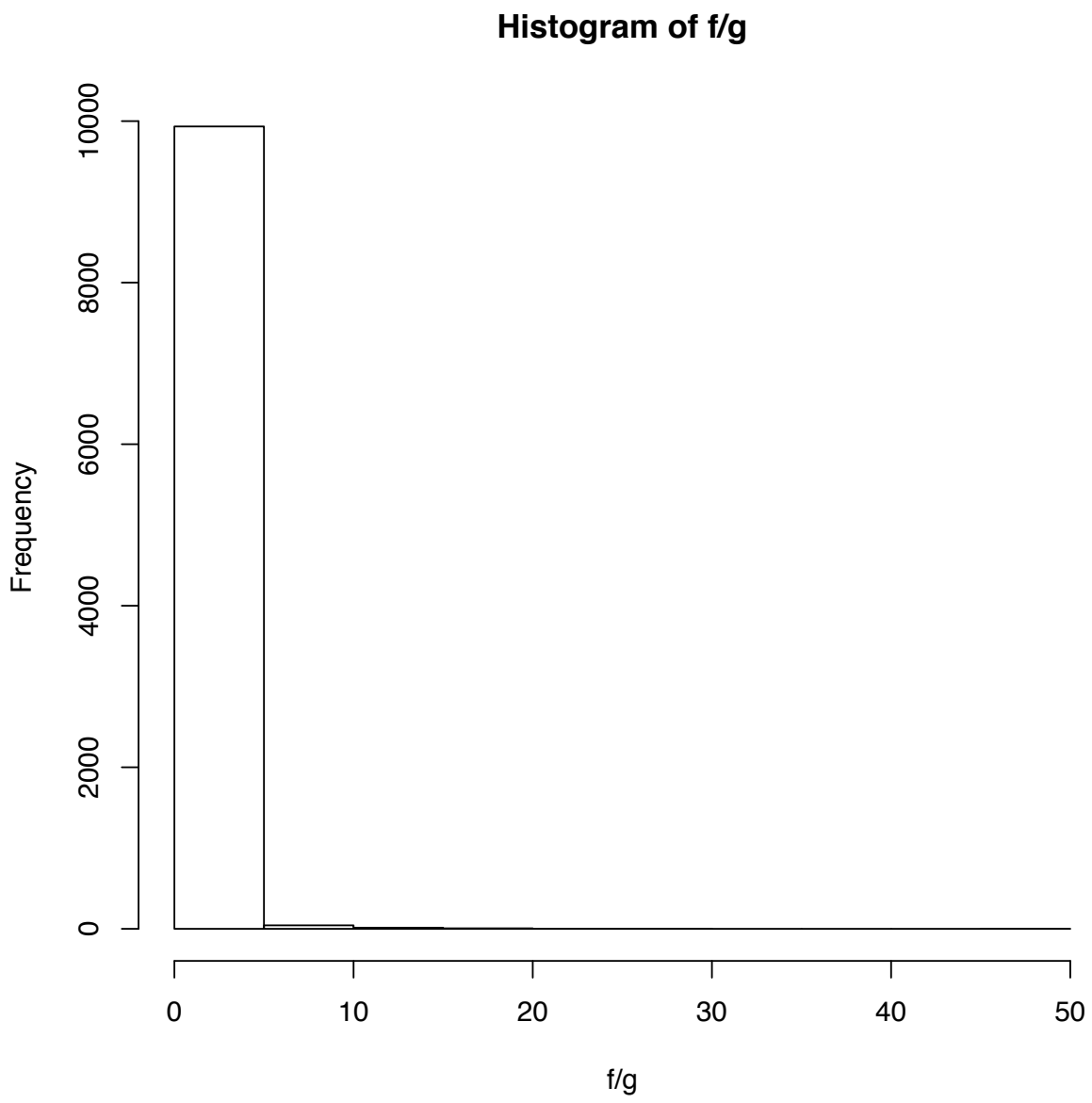
/raggedright

Question 1 (a)

```
library("truncnorm")

set.seed(0)

# Let h be the set of 10000 random numbers from truncated normal distribution
h <- rtruncnorm (n=10000, a=-Inf, b=-4, mean=-4, sd=1)

# Let f be the rescaled truncated t-distribution density composed of the random numbers
pdf <- dt(h, df=3)
cdf <- pt(-4, df=3)
f <- pdf/cdf

# Let g be the truncated normal distribution density
g <- dtruncnorm(h, b=-4, mean=-4)

# Then the importance sampling estimate of mean of h is mean(h*f/g)
estimate <- mean(h*f/g)

hist(h*f/g)
```


Histogram of h * f/g
Frequency
0
2000
4000
6000
8000
10000
−350
−300
−250
−200
−150
−100
−50
0
h * f/g

```
hist(f/g)
```

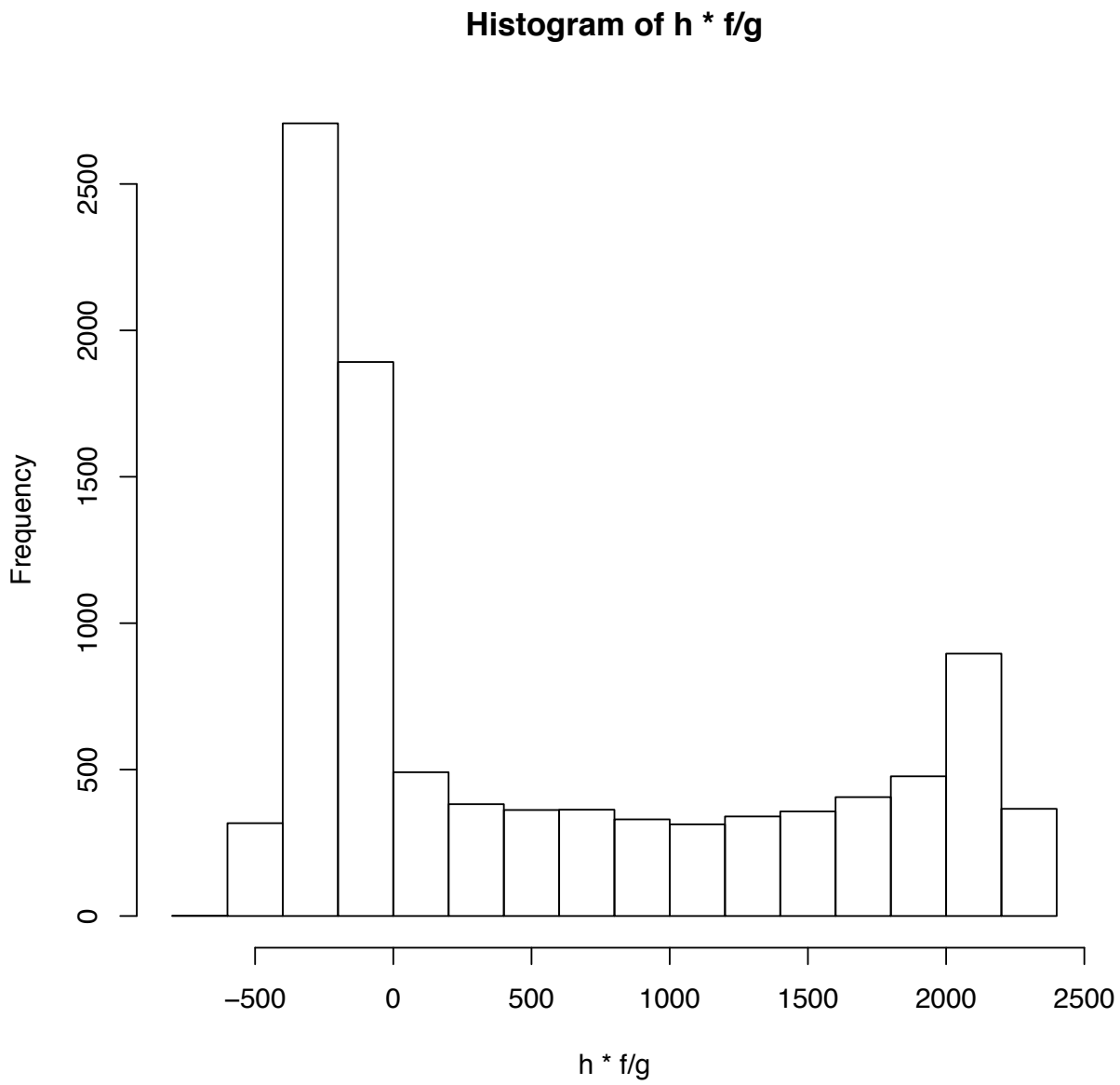



```
# We can observe that there are a lot of extreme weights
# Thus the variance of estimate is large
```

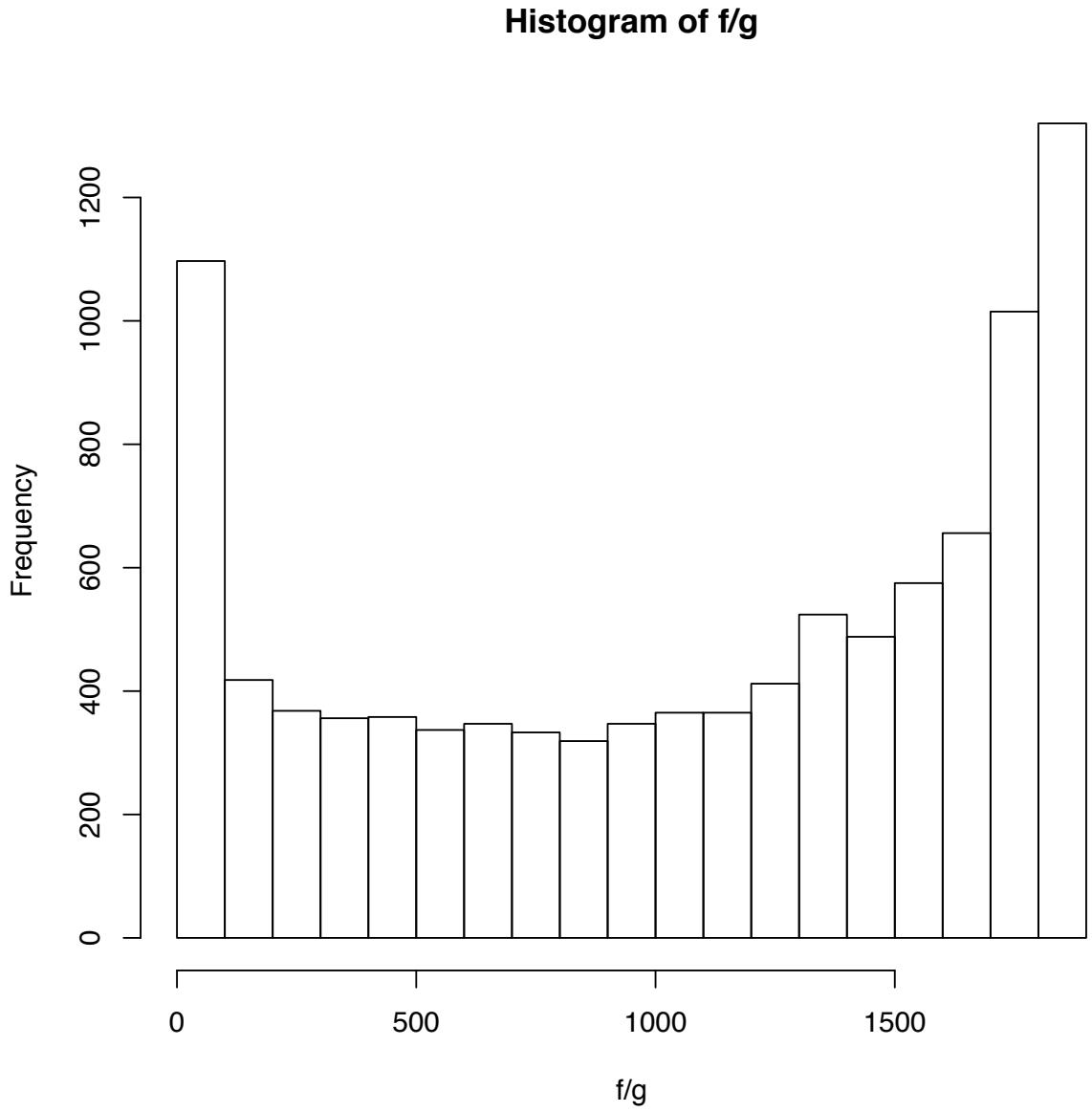
Question 1 (b)

```
set.seed(0)

# Let h be the set of 10000 random numbers from truncated t distribution
h <- rt(n=10000, df=3)
h <- -abs(x)-4

## Error:  object 'x' not found
```

```r
# Let f be the rescaled truncated t-distribution density composed of the random numbers
pdf <- dt(h, df=3)
cdf <- pt(-4, df=3)
f <- pdf/cdf

# Let g be the t-distirubtion
g <- dt(h+4, df=3)*2

# Then the importance sampling estimate of mean of h is mean(h*f/g)
estimate <- mean(h*f/g)

hist(h*f/g)
```



```r
hist(f/g)
```



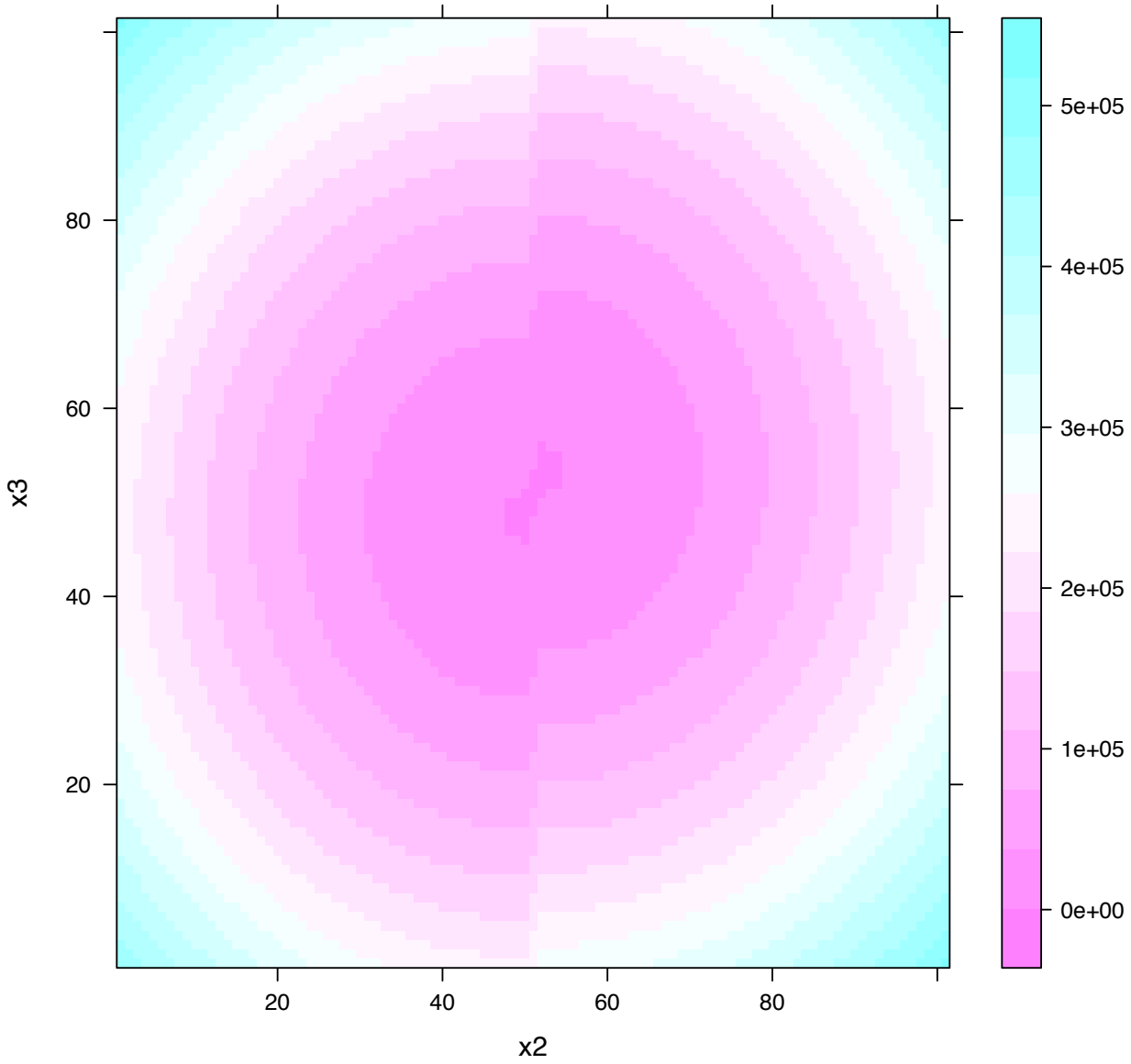

```r
# Relative to 1(a), there are less extreme weights
# Thus the variance of estimate is smaller
```

Question 2 (a)

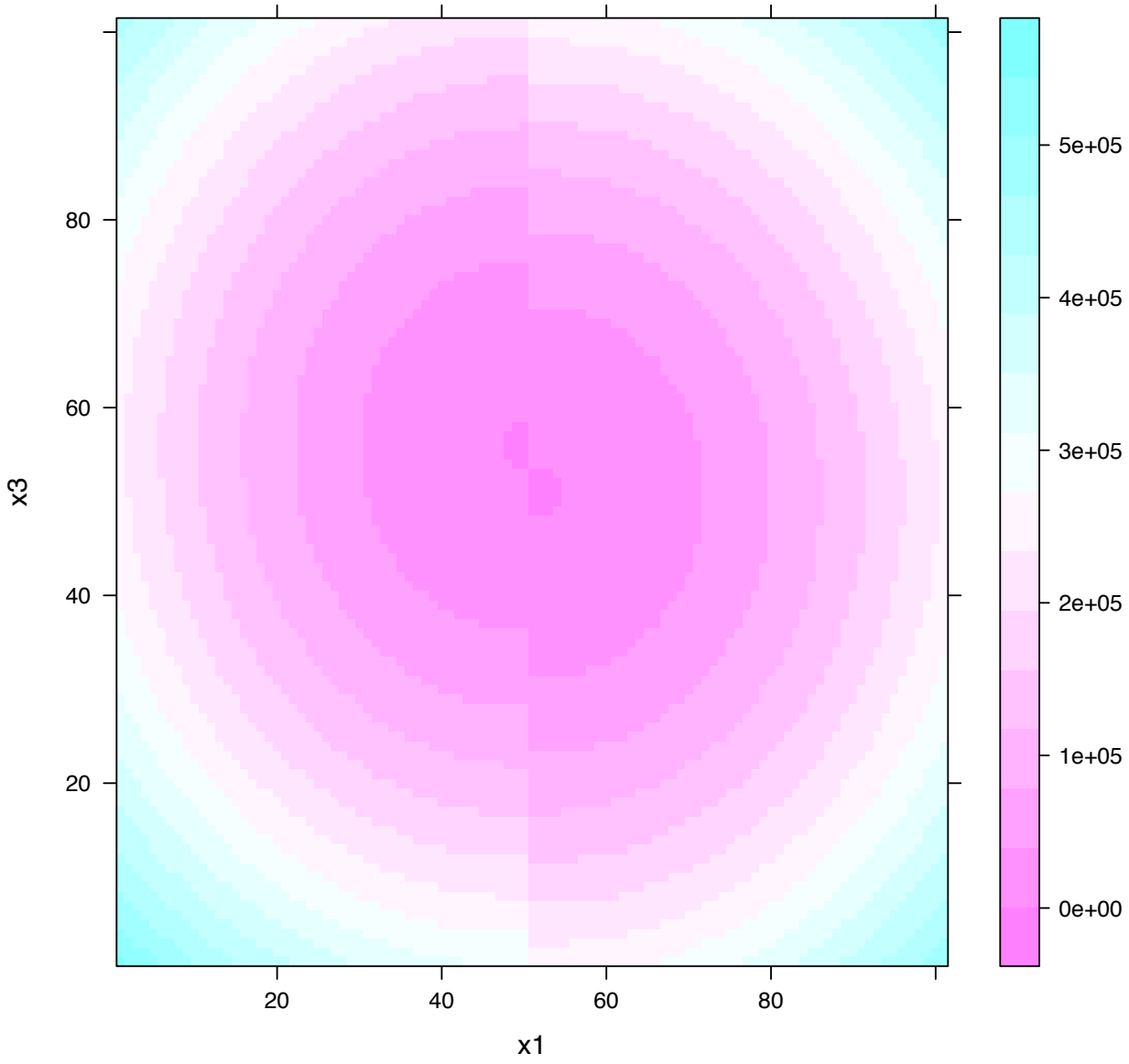
Plot slices of the function to get a sense for how it behaves.

For a constant value of one of the inputs, plot as a 2-d function of the other two.

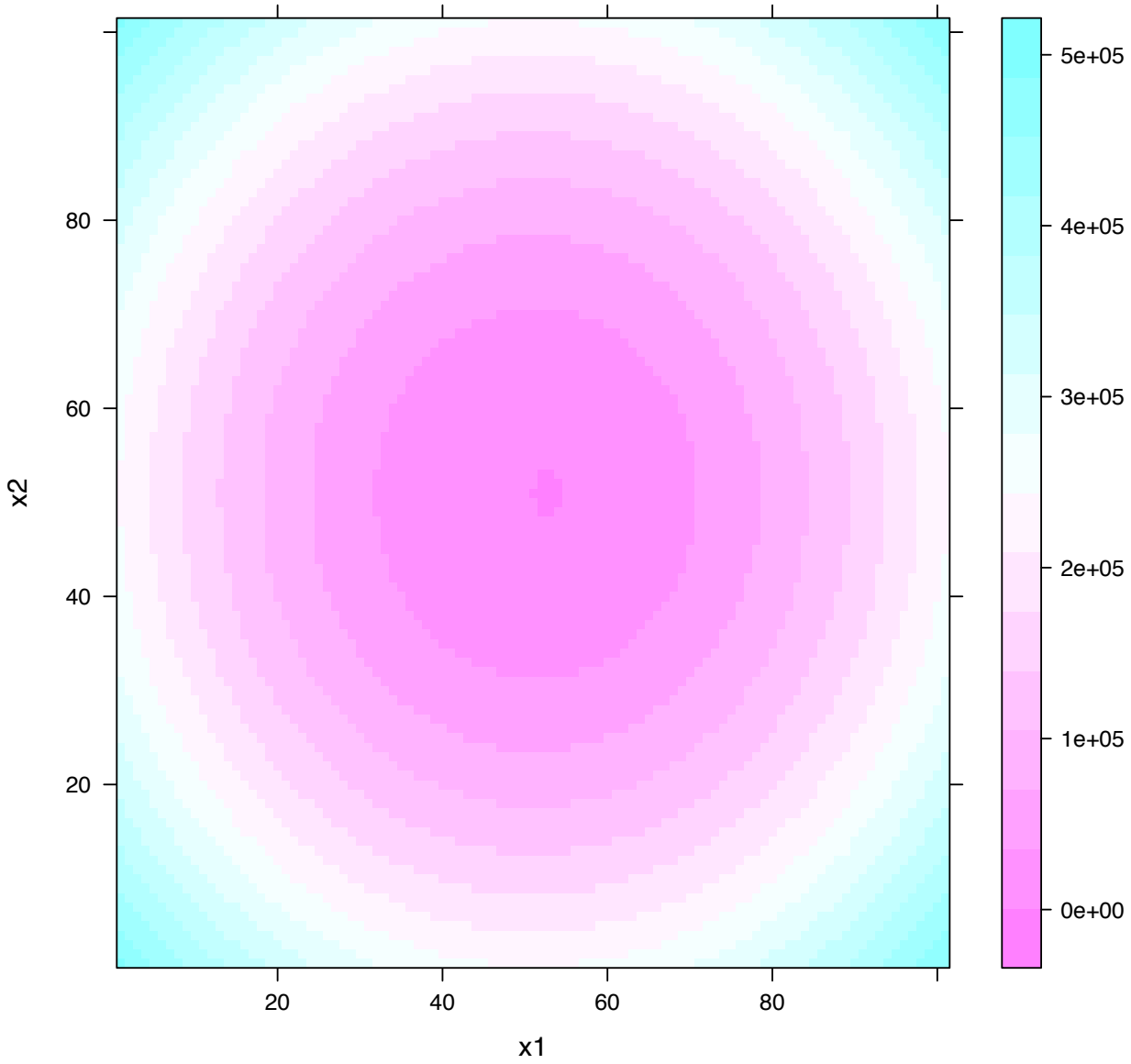
```r
# this is the given code
theta <- function(x1,x2) atan2(x2, x1)/(2*pi)

helical <- 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)
}

# set the bounds of the function we want to plot
bounds <- (-50:50)
n <- length(bounds)

# we make 3 empty matrices
matrix1 <- matrix(0,n,n)
matrix2 <- matrix(0,n,n)
matrix3 <- matrix(0,n,n)

# (i,j)th element of matrix1 will give the function output when the first input is a constant
# (i,j)th element of matrix2 will give the function output when the second input is a constant
# (i,j)th element of matrix2 will give the function output when the third input is a constant
for(i in 1:n){
  for(j in 1:n){
    matrix1[i,j] <- helical(c(0,bounds[i], bounds[j]))
    matrix2[i,j] <- helical(c(bounds[i],0,bounds[j]))
    matrix3[i,j] <- helical(c(bounds[i], bounds[j],0))
  }
}

library("reshape2")
library("lattice")

# by using melt and levelplot functions,
# we can plot the output in 2-d function of the other two
plot1 <- melt(matrix1)
names(plot1) <- c("x2", "x3", "value")
levelplot(value ~ x2 + x3, data=plot1)
```




```
plot2 <- melt(matrix2)
names(plot2) <- c("x1", "x3", "value")
levelplot(value ~ x1 + x3, data=plot2)
```




```
plot3 <- melt(matrix3)
names(plot3) <- c("x1", "x2", "value")
levelplot(value ~ x1 + x2, data=plot3)
```



Question 2 (b)

Try out optim() and nlm() for finding the minimum of this function (or use optimx()).
Explore the possibility of multiple local minima by using different starting points.

```
# first set the starting points in each row
stpts <- t(matrix(c(0,0,0,10,10,10,100,100,100),nrow=3,ncol=3))
maxit <- 10000

# the following for loop gives us the local minima using optim() and nlm() for each row
for(i in 1:nrow(stpts)){
  stpt <- stpts[i,]
```

```
  #optimize using optim()
  ctrl <- list(maxit=maxit, trace=FALSE)
  optimmin <- optim(par=stpt, fn=helical, method='BFGS', control=ctrl)

  # nlm()
  nlmmin <- nlm(f=helical, p=stpt)

 cat("starting point", stpt, "\n", "optim:", optimmin$par, "\n", "nlm():", nlmmin$estimate, "\n")
}

## starting point 0 0 0
##  optim: 1 0 0
##  nlm(): 0 0 0
## starting point 10 10 10
##  optim: 1 1.22e-08 1.944e-08
##  nlm(): 1 2.419e-10 3.472e-10
## starting point 100 100 100
##  optim: 1 -6.247e-11 -1.008e-10
##  nlm(): 1 2.38e-09 5.289e-09
```

Question 3 (a)

submitting by hand

Question 3 (b)

We can run a simple regression on the actual data that is not censored and get the whole normal distribution.
Then we can get the values of beta0, beta1, and sigma^2.
These should be reasonable starting values for those parameters.