ps8

Hangyu Huang

30 November 2017

Problem 1

(a)

lim(P(x)/f(x), x->inf)= lim(Sp/Se, x->inf) = inf

As the ratio of the survival functions diverges to infinity, the numerator(p(x)) has a heavior tail, which means the Pareto Distribution decay slowly to zero as compared to the exponential distribution.

(b)

The sampling distribution here is pareto distribution. rpareto() is used to obtain 10000 samples from the pareto distribution. There is no extreme weight that has strong influence on E(x)

```
# set 10000 random numbers from Pareto Distribution
library(EnvStats)
```

```
## Warning: package 'EnvStats' was built under R version 3.4.2
```

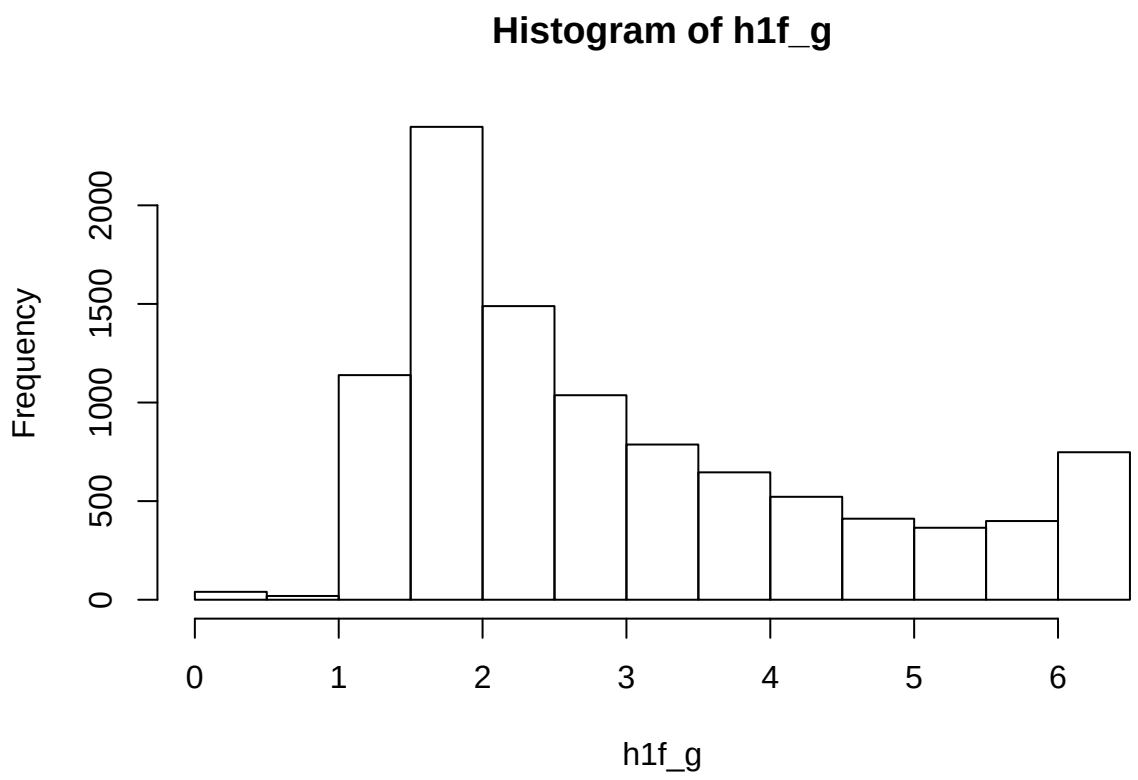
```
##
## Attaching package: 'EnvStats'
```

```
## The following objects are masked from 'package:stats':
##
##     predict, predict.lm
```

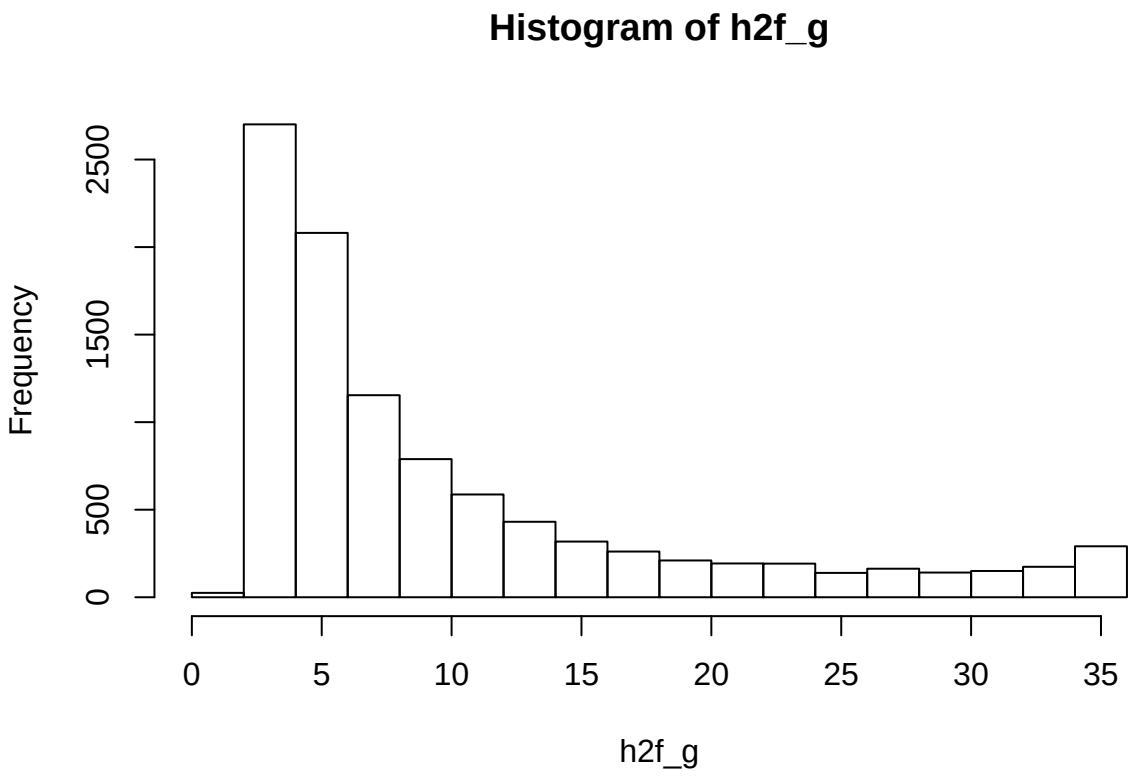
```
## The following object is masked from 'package:base':
##
##     print.default
```

```
pareto_random_numbers <- rpareto(10000,2,3)

# functions f(x),g(x),h(x)
f <- function(x, lambda = 1){
  lambda*exp(-lambda*(x-2))
}
g <- function(x, alpha=2, beta=3){
  beta*(alpha^beta)/x^(beta+1)
}
h1 <- function(x){
  x
}
h2 <- function(x){
  x^2
}

# calculation of E(x)
EX <- mean(sapply(pareto_random_numbers, FUN = function(i){
```

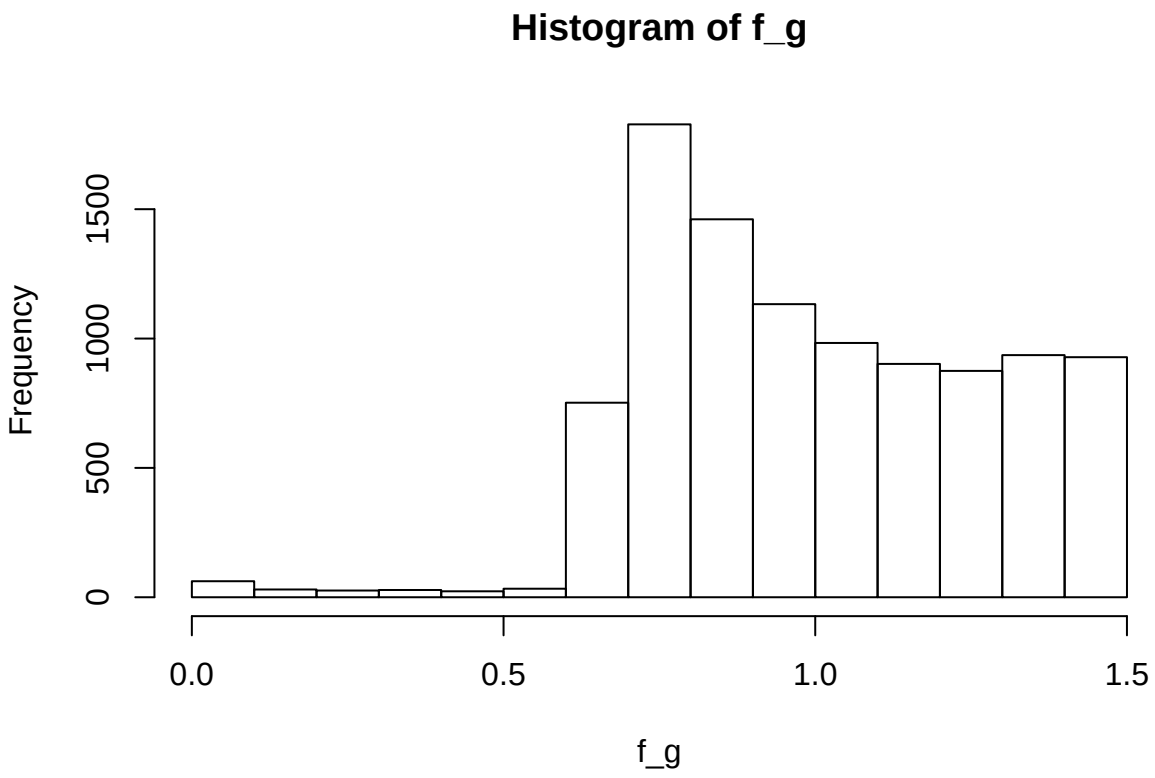
```
  h1(i)*f(i)/g(i)
}))
VX <- mean(sapply(pareto_random_numbers, FUN = function(i){
  (h1(i)*f(i)/g(i)-EX)^2
}))
# calculation of E(x^2)
X2 <- sapply(pareto_random_numbers, FUN = function(i){
  h2(i)*f(i)/g(i)
})
EX2 <- mean(X2)
VX2 <- mean(sapply(pareto_random_numbers, FUN = function(i){
  (h2(i)*f(i)/g(i)-EX2)^2
}))
# histogram of h(x)f(x)/g(x)
h1f_g <- sapply(pareto_random_numbers, FUN = function(i){
  h1(i)*f(i)/g(i)
})
hist(h1f_g)
```




```
h2f_g <- sapply(pareto_random_numbers, FUN = function(i){
  h2(i)*f(i)/g(i)
})
hist(h2f_g)
```




```
# histogram of f(x)/g(x)
f_g <- sapply(pareto_random_numbers, FUN = function(i){
  f(i)/g(i)
})
hist(f_g)
```

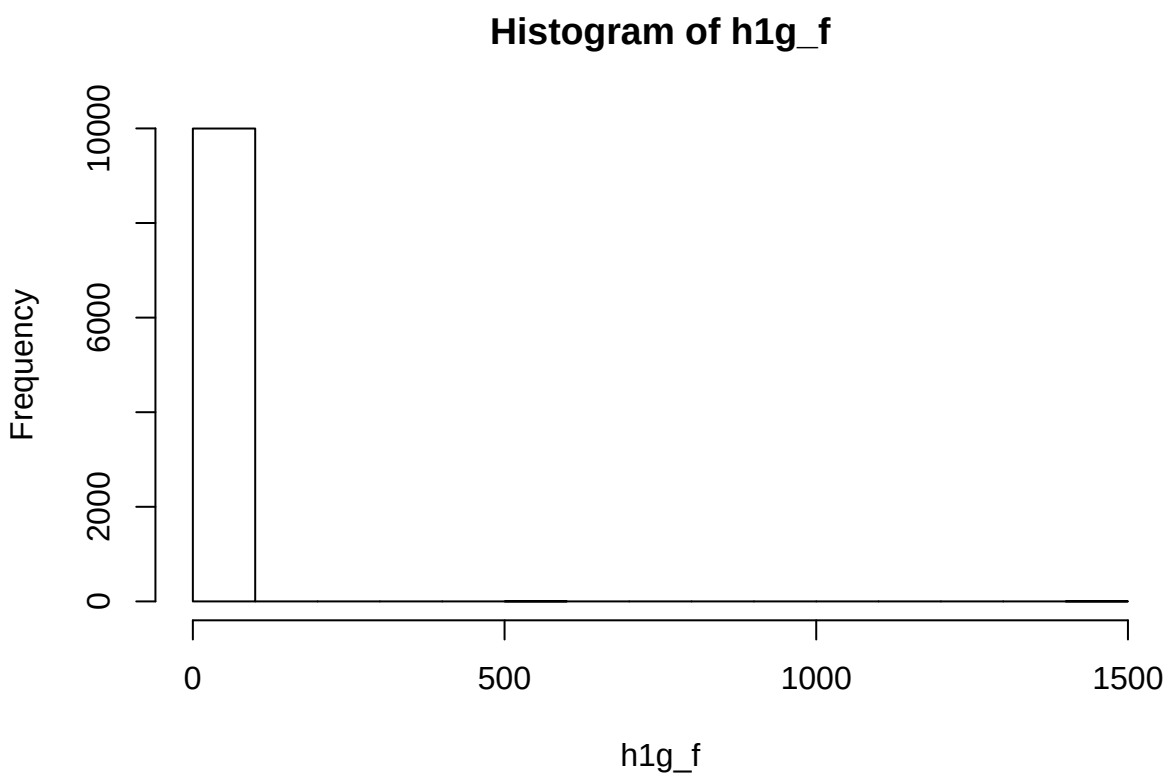


1(c)

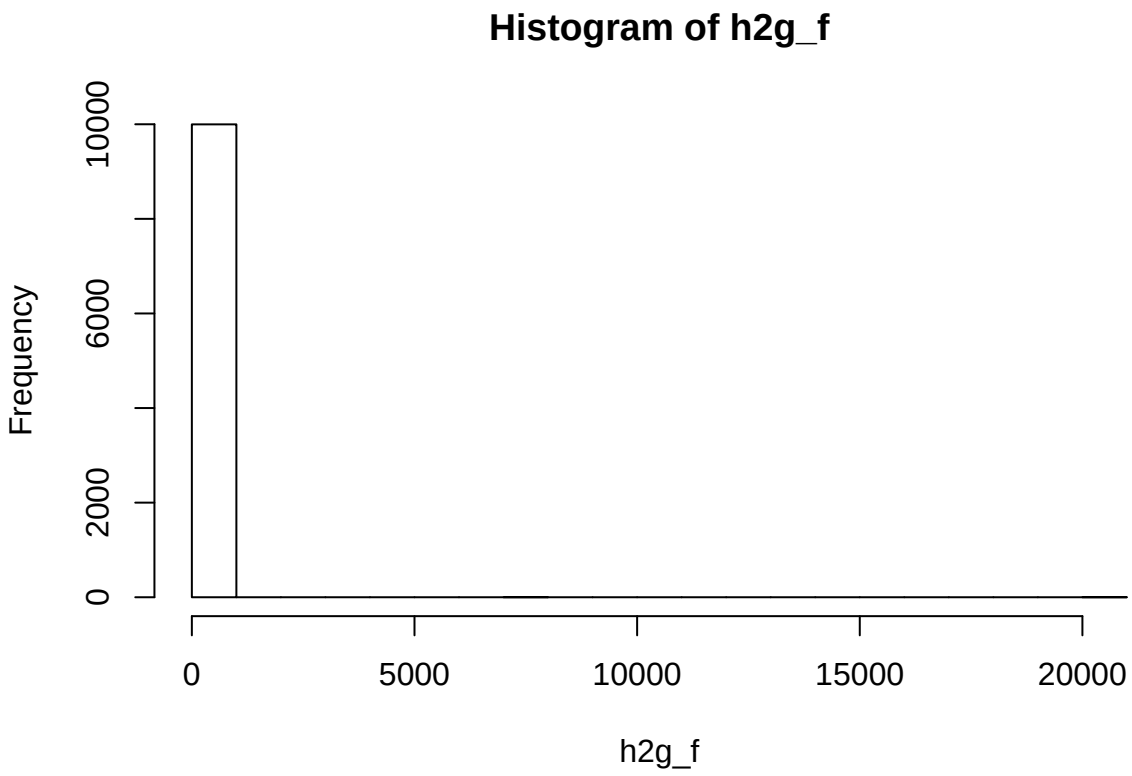
The sampling distribution is exponential dirstribution. rexp() is used to find 10000 random numbers from the exponential distribution. There is no extreme weight that has strong influence on E(x). The Var(phi) in condition(b) is much smaller than that in conidition(c)

```
# random numbers
exp_random_numbers <- rexp(10000,1)
exp_random_numbers <- exp_random_numbers + 2
# calculation of E(x)
EX <- mean(sapply(exp_random_numbers, FUN = function(i){
  h1(i)*g(i)/f(i)
}))
VX <- mean(sapply(exp_random_numbers, FUN = function(i){
  (h1(i)*g(i)/f(i)-EX)^2
}))
# calculation of E(x^2)
X2 <- sapply(exp_random_numbers, FUN = function(i){
  h2(i)*g(i)/f(i)
})
EX2 <- mean(X2)
VX2 <- mean(sapply(exp_random_numbers, FUN = function(i){
  (h2(i)*g(i)/f(i)-EX2)^2
}))
# histogram of h(x)f(x)/g(x)
```

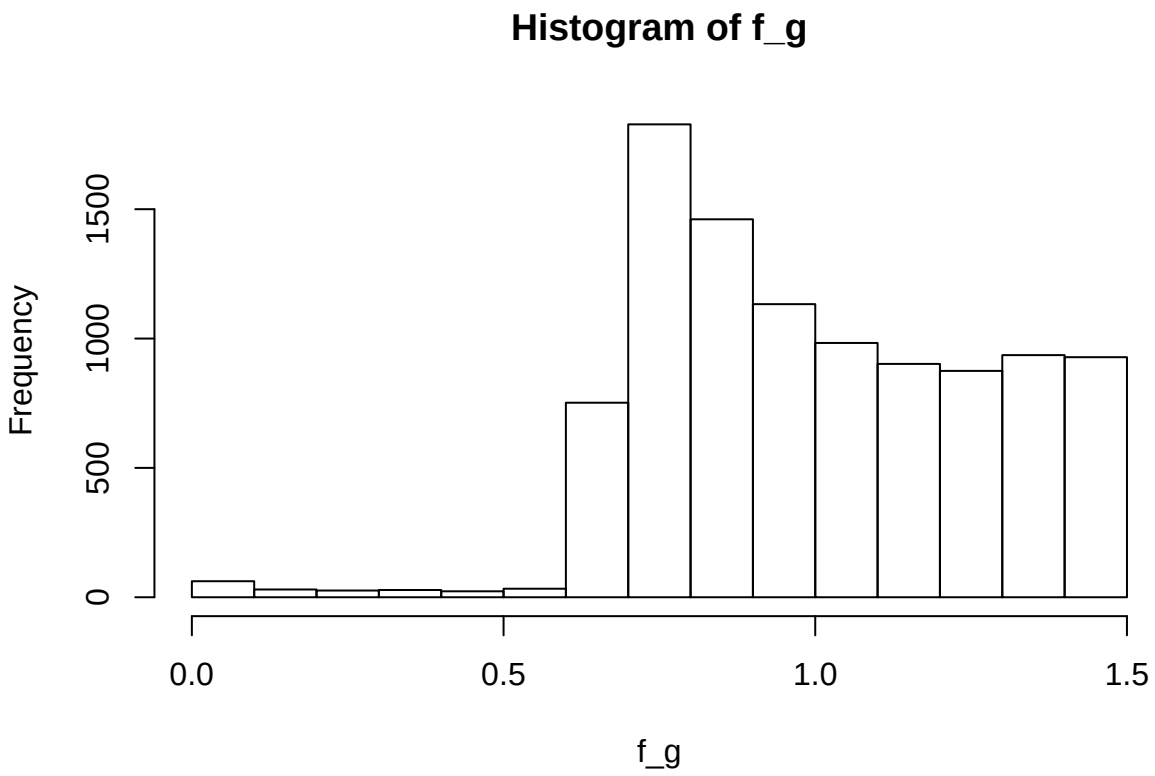
```r
h1g_f <- sapply(exp_random_numbers, FUN = function(i){
  h1(i)*g(i)/f(i)
})
hist(h1g_f)
```




```r
h2g_f <- sapply(exp_random_numbers, FUN = function(i){
  h2(i)*g(i)/f(i)
})
hist(h2g_f)
```




```
# histogram of f(x)/g(x)
g_f <- sapply(exp_random_numbers, FUN = function(i){
  g(i)/f(i)
})
hist(f_g)
```



Problem 2 first, we set x3=0, x1 is (-5,5), x2 is (-10, 10). we use image.plot to plot a image with x and y. and we find the minimum of this function is almost at (1,0,0) by nlm() second, we set x1=0, x2 is (-10,10), x3 9s (-10,10).we use image.plot to plot a image with y and Z. and we find the minimum of this function is almost at (1,0,0) by nlm()

```
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)
}


## explore surface {at x3 = 0}
library(graphics)
library(stats)
library(utils)
require(fields)
```

```
## Loading required package: fields
```

```
## Loading required package: spam
```

```
## Loading required package: dotCall64
```

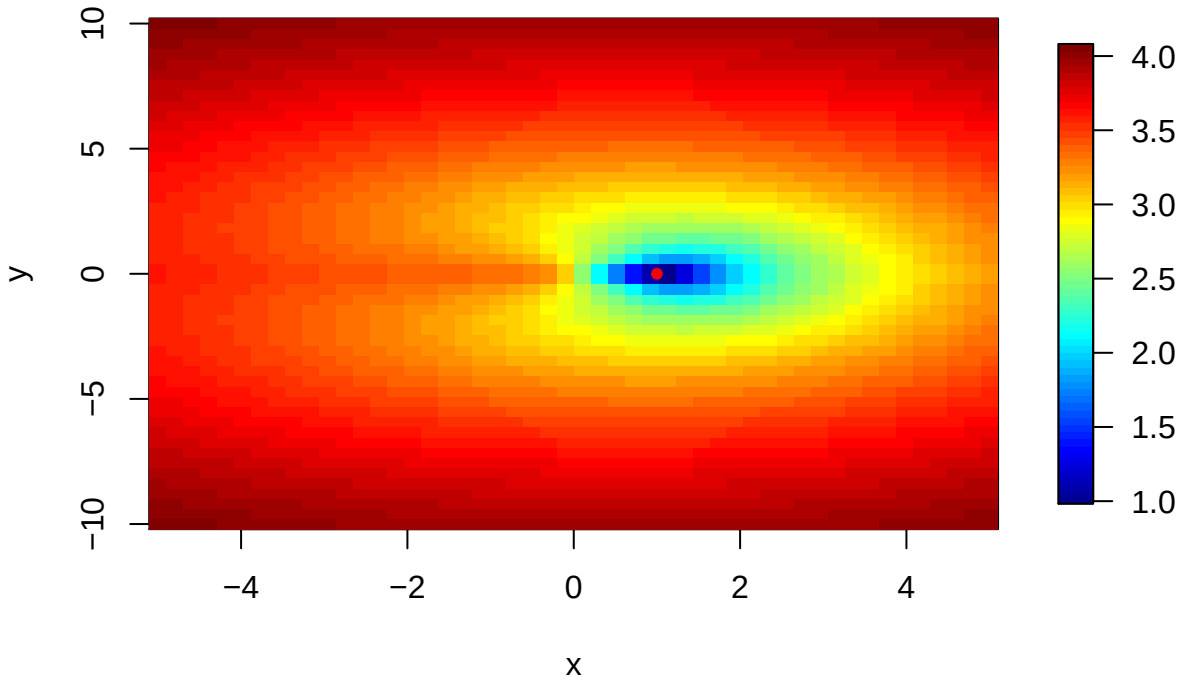
```
## Loading required package: grid
```

```
## Spam version 2.1-1 (2017-07-02) is loaded.
## Type 'help( Spam)' or 'demo( spam)' for a short introduction
## and overview of this package.
## Help for individual functions is also obtained by adding the
## suffix '.spam' to the function name, e.g. 'help( chol.spam)'.

##
## Attaching package: 'spam'

## The following objects are masked from 'package:base':
##
##     backsolve, forwardsolve

## Loading required package: maps
```

```
x <- seq(-5, 5, length.out=50)
y <- seq(-10, 10, length.out=50)
z <- apply(as.matrix(expand.grid(x, y)), 1, function(x) f(c(x, 0)))
image.plot(x, y, matrix(log10(z), 50, 50))
str(nlm.f <- nlm(f, c(-1,0,0), hessian = TRUE))
```

```
## List of 6
##  $ minimum   : num 1.24e-14
##  $ estimate  : num [1:3] 1.00 3.07e-09 -6.06e-09
##  $ gradient  : num [1:3] -3.76e-07 3.49e-06 -2.20e-06
##  $ hessian   : num [1:3, 1:3] 2.00e+02 -4.07e-02 9.77e-07 -4.07e-02 5.07e+02 ...
##  $ code      : int 2
##  $ iterations: int 27
```

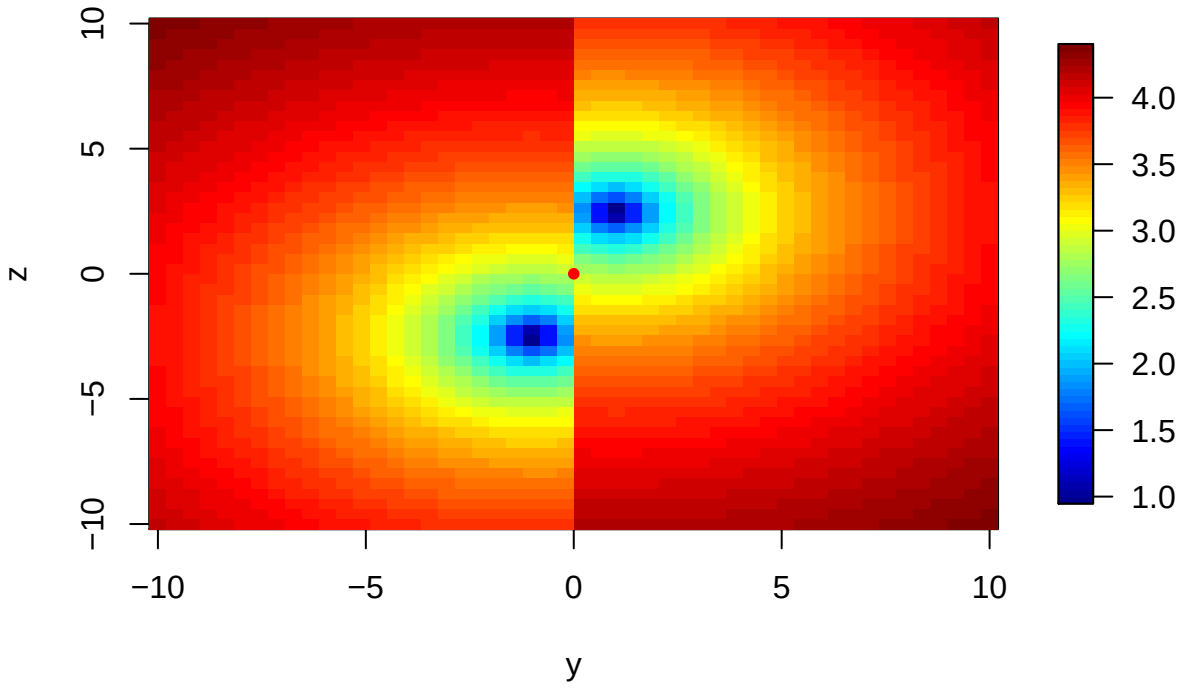
```
points(rbind(nlm.f$estim[1:2]), col = "red", pch = 20)
```




```
stopifnot(all.equal(nlm.f$estimate, c(1, 0, 0)))

## explore surface {at x1 = 0}
y <- seq(-10, 10, length.out=50)
z <- seq(-10, 10, length.out=50)
x <- apply(as.matrix(expand.grid(y, z)), 1, function(x) f(c(0,x)))
image.plot(y, z, matrix(log10(x), 50, 50))
str(nlm.f <- nlm(f, c(-5,-5,0), hessian = TRUE))
```

```
## List of 6
##  $ minimum   : num 1.68e-17
##  $ estimate  : num [1:3] 1.00 1.88e-09 3.23e-09
##  $ gradient  : num [1:3] 2.27e-08 -7.21e-08 5.17e-08
##  $ hessian   : num [1:3, 1:3] 2.00e+02 -4.07e-02 6.00e-07 -4.07e-02 5.07e+02 ...
##  $ code      : int 1
##  $ iterations: int 30
```

```
points(rbind(nlm.f$estim[2:3]), col = "red", pch = 20)
```




```
stopifnot(all.equal(nlm.f$estimate, c(1, 0, 0)))
```

problem 3

3a

1. the linear regression model: As W???N(??,??2),the conditional distribution of Y given W is W~N(beta0+beta1X,sigma^2). Then we can write down the joint probability density of W and Y given by

f(yi,wi)=f(yi|wi)f(wi)

the log-likelihood for the model(the obeserved part and the missing part)

L(theta;X,Y)=s(i=1~c)L(thata;wi,yi)

2. E step: find the expection of missing values

Xcomp=(x1,.,xm,xm+1,.,xc)=(Xobs,Xmis)

then the complete data log-likelihood for the modal is decomposed into the observed and the missing part

M1 = E(W|W>Tau); M2 = V(W|W>Tau)-M1^2 Q(Theta,Theta*)=L(Theta;W,Y)

3. M step : the maximum Q(Theta,Theta*) calcu-lated in the E-step is when its derivation of theta is zero. After solving this equation, we can find the updated estimates

3b

the starting values are the quantiles at problility = 20%.

3c

```
#E-step
M <- function(Mu,Sigma,Tau) {
  Tau1 <- (Tau-Mu)/Sigma
  Rho <- dnorm(Tau1, mean =0, sd = 1)/(1-pnorm(Tau1, mean = 0, sd = 1))
  M1 <- Mu + Sigma*Rho
  V <- Sigma^2 * (1+Tau1*Rho-Rho^2)
  M2 <- V - M1
  return(M1,M2)
}

#M-step

updatedTheta <- function(Xbs,M1, M2, y,beta0, beta1, sigma2){
  sumof

}

#data provided
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
```