Stat-6207 Homework Set 3 Fall 2018(Modarres)

1) To measure the breaking strength of material Swedish physicist Waloddi Weibul proposed the Weibul distribution with density function $f(x) = \lambda x^{\lambda-1}\exp(-x^{\lambda})$ for $x, \lambda > 0$. Suppose you have a random sample $x_1, \ldots, x_n$ from the Weibul distribution and would like to estimate the maximum likelihood of λ.

- a. Find the likelihood function $L(\lambda|x_1, \ldots, x_n)$ and the log-likelihood function.
- b. Find the score functions $S(\lambda) = \frac{\partial \ln(L(\lambda|x_1,\ldots,x_n))}{\partial \lambda}$ and its derivative.
- c. Using Newton's method, write an R function for the MLE of λ when $x = (0.1, 0.25, 0.5, 1, 2)$.

2) Insert keys $(45, 24, 13, 0, 7, 23, 30, 43, 32)$ into a binary search tree where each node is a list of 4 fields: (left pointer,key, rank, right pointer). The rank of a node is one plus the number of nodes in its left subtree.

- a. Show the tree with nodes composed of value and rank field.
- b. Show the inorder traversal of the tree.
- c. Show the sequence of calls to find the median, minimum and maximum.
- d. Let $S[x]$ be the number of nodes in a binary search tree rooted at x. Find a recursive expression for $S[x]$.

3) Suppose the postorder traversal of a binary tree gives $(A, B, G, K, H, F, Y, Z, X, R, M)$ whereas its inorder traversal gives $(A, B, F, G, H, K, M, R, X, Y, Z)$. Draw the binary tree and give its preorder traversal.

4) Obtain the time complexity of the following code segments in terms of N.

- a. $for\ \ (i\ in\ 1:N)\{for\ \ (j\ in\ 1:i)\ print(x)\}$
- b. $f2 = function(N)\{if(N == 0)return(1); x = 0; for\ \ (i\ in\ 1:N)\ x = x + f2(N-1)\}$
- c. $f3 = function(N)\{if(N == 1)return(0); return(1 + f3(N/2))\}$

5) When a data set is too large to be stored in primary memory, standard algorithms for computing sample quantiles are not directly applicable. You will need to write a function for estimating the sample median of n observations in m memory locations using n random numbers from a standard normal distribution. To describe the method of recursive medians, suppose that m is of the form b^k where b and k are integers. The **remedian algorithm** processes the observations sequentially in groups of size b. A median is computed for each group, yielding b^{k-1} medians in the first stage. This step is repeated recursively until a single estimate is found. A median can be computed using k arrays of size b requiring $m = kb$ storage locations. Let $b = 15$ and $k = 4$ so that $n = 50,625$ and $m = 60$. Replicate your result 100 times. Obtain univariate summary statistics and compare to the true median. See the paper on the Remedian.

6) We are given n objects in a knapsack. Object i has weight w_i and the knapsack has capacity M. If object i is placed into the knapsack, then a profit of p_i is earned. The objective is fill the knapsack to maximize profit. Hence we want to maximize $\sum_{i=1}^{n} p_i x_i$ subject to $\sum_{i=1}^{n} w_i x_i \leq M$ and $x_i = 0, 1$ for $i = 1, \ldots, n$.

- a. Consider the following instance of the knapsack problem: $n = 3$, $M = 20$, $P = (p_1, p_2, p_3) = (25, 24, 15)$, and $W = (w_1, w_2, w_3) = (18, 15, 10)$. By hand calculations find the feasible solutions and obtain the optimal selection when $X = (x_1, x_2, x_3)$ is i) $(1, 0, 1)$, ii) $(1, 0, 0)$, iii) $(0, 0, 1)$, and iv) $(1, 1, 1)$.

- b. We would like to devise a randomized algorithm to find a suboptimal solution. Suppose $P = (seq(1, 20) - 10.5)^2/(10 + seq(1, 20))$ and $w = seq(1, 20)/(1 + seq(1, 20))$. Write an R function that finds a randomized solution by generating 1000 instances vector X with independent components $Bernoulli(0.5)$. Let $M = 15$.

Q 1) a) $L(\lambda \mid x_1, \dots, x_n) = \lambda^n \left(\prod_{i=1}^{n} x_i^{\lambda-1} \right) e^{-\sum_{i=1}^{n} x_i^{\lambda}}$

$\ln(L(\lambda \mid x_1, \dots, x_n)) = n \ln \lambda + (\lambda - 1) \sum_{i=1}^{n} \ln x_i - \sum_{i=1}^{n} x_i^{\lambda}$

b) $S(\lambda) = \frac{\partial \ln(L(\lambda \mid x_1 \dots x_n))}{\partial \lambda} = \frac{n}{\lambda} + \sum_{i=1}^{n} x_i - \sum_{i=1}^{n} x_i^{\lambda} \ln x_i$

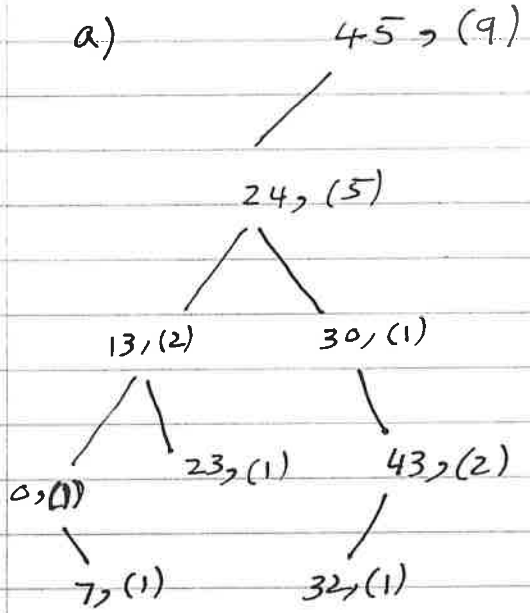
$\frac{\partial S(\lambda)}{\partial \lambda} = \frac{-n}{\lambda^2} - \sum_{i=1}^{n} x_i^{\lambda} (\ln x_i)^2$

c) see plot

d) see R code

```
# Q1 HW3 Stat 6207

# Find MLE of Weibul f(x)=lam x^(lam-1) exp(-x^lam)) where x,lam>0.

Weibulmle<-function(x,lam){
  #Score function for the MLE of Lam

  n/lam+sum(log(x))-sum((x^lam)*log(x))
}

sprime<-function(x,lam){
   #The derivative of the  score function.
  -n/lam^2-sum(x^lam*(log(x))^2)
}

newton<-function(x,fn,fprime,start,tol=1e-10){
  #This function performs the univariate Newton's method to find the root of an
equation.
  i<-0
  cur<-start
  res<-c(i,cur,fn(x,cur))
  while (abs(fn(x,cur))>tol) {
    i<-i+1
    cur<-cur-fn(x,cur)/fprime(x,cur)
    res<-rbind(res,c(i,cur,fn(x,cur)))
    show(res)
  }
  return(res)
}

x<-c(.1,.25,.5,1,2)
xbar=mean(x)
n=5
val=seq(1,100)/50
ww=newton(x, Weibulmle,sprime,mean(x),tol=10e-16)
lamle=ww[5,2]
lamle


res     0 0.770000  2.896859e+00
        1 1.031085  6.293860e-01
        2 1.122649  3.988779e-02
        3 1.129254  1.729451e-04
        4 1.129283  3.267475e-09
        5 1.129283 -1.110223e-16
> lamle=ww[5,2]
> lamle

1.129283
>
```

2) a)



b) 0, 7, 13, 23, 24, 30, 32, 43, 45

c) Rank(Median) = 5 ⇒ Go left from 45 ⇒ median = 24

Rank(Min) = 1 ⇒ Go left, Go left, Go left ⇒ Min = 0

Rank (Max) = 9 ⇒ Max = 9 match with The root.

d) size [x] = size [Left [x]] + size [righ [x]] + 1

with size [NiL] = 0

For example, size [24] = size [left [24]] + size [right [24]] + 1

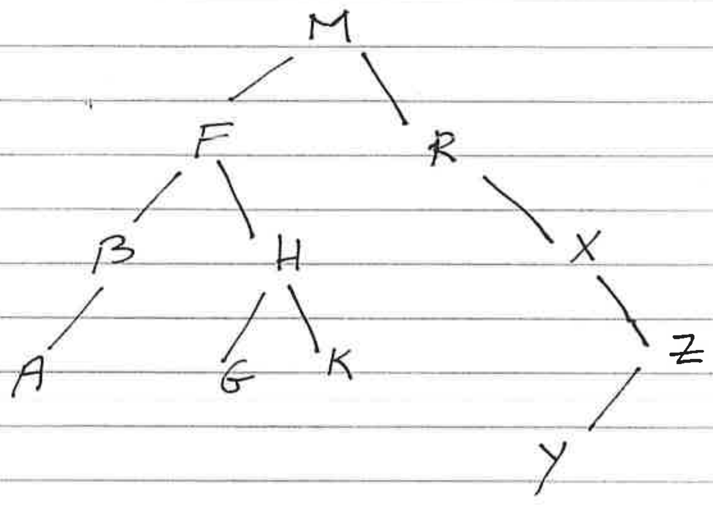
= 4 + 3 + 1 = 7

4) a) $\sum_{i=1}^{N} \sum_{j=1}^{i} 1 = \sum_{i=1}^{N} (i-1+1) = \frac{N(N+1)}{2} \Rightarrow O(N^2)$

b) $N(N-1)(N-2)\ldots 3\cdot 2\cdot 1 = N! \Rightarrow O(N!)$

c) $O(\log_2(N))$ sin6 $\frac{N}{2^K} = 1 \Rightarrow K = \log_2 N$

3)



In order A B F G H K M R X Y Z

past order A B G K H F Y Z X R M

preorder M F B A H G K R X Z Y

```
#  Q5 HW3 Stat 6207

############################## RM ###########################
rm <- function(b){
  a1=rep(b,1)
  a2=a1
  a3=a1
  a4=a1
  high=0
  for(k1 in (1:b)) {
    for(k2 in (1:b)) {
      for(k3 in (1:b)) {
        low=high+1
        high=high+b
        a1=x[low:high]
        a2[k3]=median(a1)
      }
      a3[k2]=median(a2)
    }
    a4[k1]=median(a3)
  }
  m=median(a4)
  return(m)
}
################# Main ####################
rept=100
n=50625
b=15
k=4
m=b*k
remed=rep(0,rept)
truemed=rep(0,rept)
for (i in (1:rept)) {
    x=rnorm(n)
    truemed[i]=median(x)
    remed[i]=rm(b)
}
summary(remed)
summary(truemed)
summary(remed)
      Min.       1st Qu.     Median        Mean       3rd Qu.     Max.
      -0.0380300  -0.0073320  0.0009776    -0.0002721  0.0072520   0.0207300
> summary(truemed)
         Min.    1st Qu.     Median        Mean       3rd Qu.     Max.
      -0.0163400  -0.0037540  0.0003048    -0.0003207  0.0033520   0.0125200
```

```
#Question 6  Stat 6207

# P[1:n] and W[1:n] contain the profits and weights
# respectively of n objects.
# M is the knapsack size and X[1:n] is the solution vector.

Knapsack <- function(P,W,M,X,n) {
  sumwx=t(W)%*%X
  sumpx=t(P)%*%X
  cat('sumWX= ',sumwx,'sumPX= ',sumpx)
}
n=3
M=20
P=c(25,24,15)
W=c(18,15,10)
X1=c(1/2,1/3,1/4)
X2=c(1,2/15,0)
X3=c(0,2/3,1)
X4=c(0,1,1/2)
Knapsack(P,W,M,X1,n)

sumWX=  16.5 sumPX=  24.25

Knapsack(P,W,M,X2,n)

sumWX=  20 sumPX=  28.2

Knapsack(P,W,M,X3,n)

sumWX=  20 sumPX=  31

Knapsack(P,W,M,X4,n)

sumWX=  20 sumPX=  31.5      ←  Max profit

n=20
P=(seq(1,n)-10.5)^2/(10+seq(1,n))
W=seq(1,n)/(1+seq(1,n))
rep=1000
M=15
maxpx=-1;
for (i in 1:rep) {
   X=rbinom(n,1,.5)
   sumwx=t(W)%*%X
   if (sumwx<=M){
      sumpx=t(P)%*%X
      if (sumpx>maxpx) {
         maxpx=sumpx
         xvec=X}
   }
}
cat('max= ', maxpx,'with sumwx= ',sumwx, 'at X= ',X,fill=TRUE)
max=  35.99621 with sumwx=  9.02417 at X=  1 1 1 1 1 1 0 1 0 1 1 0
0 0 0 0 0 0 1 1
```