11111

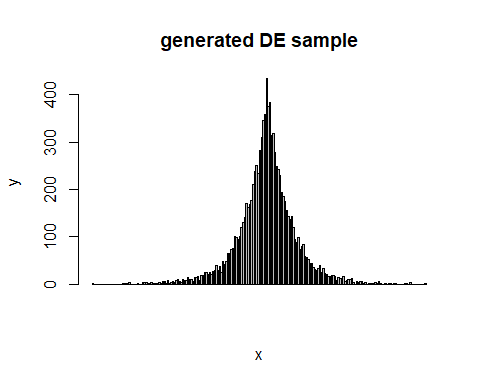
Wuhao Wang

11/22/2021

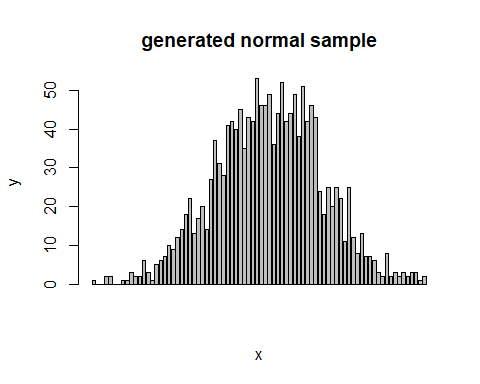
# Question 2   
# q1  
unif\_sample <- runif(10000,0,1)  
DE\_sample <- c()  
for (i in 1:10000)  
{  
 if(unif\_sample[i] > 0.5)  
 DE\_sample <-c(DE\_sample,log(2-2\*unif\_sample[i]))  
 else  
 DE\_sample <-c(DE\_sample,-log(2\*unif\_sample[i]))  
}  
mean(DE\_sample)

## [1] -0.008382104

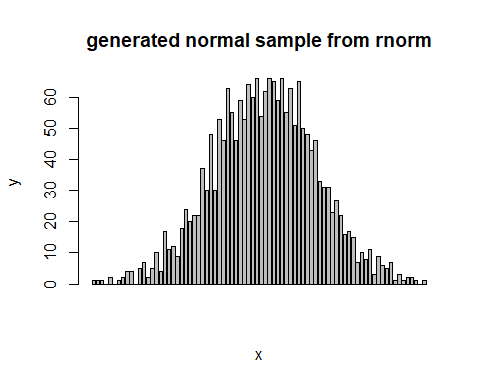
# interval [n\*gap,(n+1)\*gap] will be used in the histogram  
# the value of gap is 0.0986664214901679  
gap <- (max(DE\_sample)-min(DE\_sample))/200  
# count will be vector used to make histogram  
count <-c()  
start <- min(DE\_sample)  
end <- min(DE\_sample) + gap  
for(i in 1:200)   
{   
 temp <- DE\_sample[which(DE\_sample >= start)]  
 temp <- temp[which(temp < end)]  
 count <- c(count,length(temp))  
 start <- start + gap  
 end <- end + gap  
}  
barplot(count,xlab = 'x',ylab = 'y',main = 'generated DE sample')



rm(start,end,count,gap,unif\_sample,i,temp)  
################################################################################  
# q2  
#  
c = 1.3155  
index <- 1:10000  
DE\_2000 <- DE\_sample[which(index%%5==0)]  
reject\_num <- 0  
accept\_sample <- c()  
#for(j in 1:300){  
for (i in 1:2000)   
{  
 y <- DE\_2000[i]  
 # fxx is normal density at x  
 u <- runif(1)  
 fxy <- dnorm(y)  
 fyy <- 1/2 \*exp(-abs(y))  
 if(u <= fxy/(fyy\*c))  
 {accept\_sample <- c(accept\_sample,y)}  
 else  
 {  
 reject\_num <- reject\_num + 1  
 }  
}  
#}  
#reject\_num/300/2000  
# ER is 1-1/c  
gap <- (max(accept\_sample)-min(accept\_sample))/80  
# count will be vector used to make histogram  
count <-c()  
start <- min(accept\_sample)  
end <- min(accept\_sample) + gap  
for(i in 1:80)   
{   
 temp <- accept\_sample[which(accept\_sample >= start)]  
 temp <- temp[which(temp < end)]  
 count <- c(count,length(temp))  
 start <- start + gap  
 end <- end + gap  
}  
barplot(count,xlab = 'x',ylab = 'y',main = 'generated normal sample')



r<-rnorm(2000)  
# ER is 1-1/c  
gap <- (max(r)-min(r))/80  
# count will be vector used to make histogram  
count <-c()  
start <- min(accept\_sample)  
end <- min(accept\_sample) + gap  
for(i in 1:80)   
{   
 temp <- r[which(r >= start)]  
 temp <- temp[which(temp < end)]  
 count <- c(count,length(temp))  
 start <- start + gap  
 end <- end + gap  
}  
barplot(count,xlab = 'x',ylab = 'y',main = 'generated normal sample from rnorm')  
################################################################################  
################################################################################  
################################################################################  
#Question 1  
#q1  
library(ggplot2)



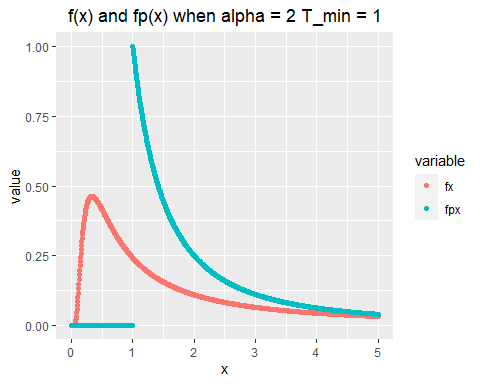
library(reshape2)

## Warning: package 'reshape2' was built under R version 4.1.1

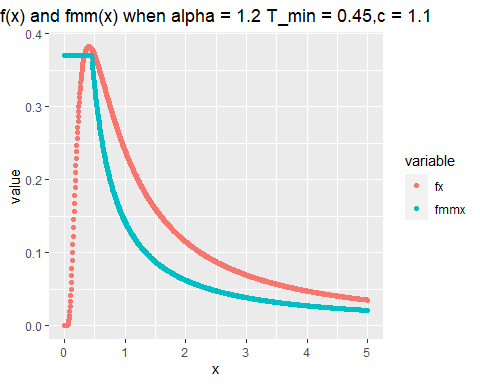
library(poweRlaw)

## Warning: package 'poweRlaw' was built under R version 4.1.1

get\_fpx\_origin <- function(alpha,t\_min,x)  
{  
 x <- x[order(x)]  
 s <- x[which(x<t\_min)]  
 x <- x[-which(x<t\_min)]  
 res <- rep(0,length(s))  
 res <- c(res,(alpha-1)/t\_min\*(x/t\_min)^(-alpha))  
 return(res)  
}  
get\_fpx <- function(alpha,t\_min,x)  
{  
 x <- x[order(x)]  
 s <- x[which(x<t\_min)]  
 x <- x[-which(x<t\_min)]  
 res <- rep((alpha-1)/t\_min,length(s))  
 res <- c(res,(alpha-1)/t\_min\*(x/t\_min)^(-alpha))  
 res <- res/alpha  
 return(res)  
}  
# plot for f(x) and fp(x)  
c <- 1  
t\_min <- 1  
alpha <- 2  
x\_f <- 1:1000/200  
fx <- c/sqrt(2\*pi)\*exp(-c\*c/2/x\_f)\*x\_f^(-3/2)  
fpx <- get\_fpx\_origin(alpha,t\_min,x\_f)  
df <- data.frame(x = x\_f,fx =fx,fpx=fpx)  
df1 <- melt(df,id.vars='x')  
  
p1 <-ggplot(df1,aes(x=x,y=value))+  
 geom\_point(aes(color=variable))+  
 ggtitle('f(x) and fp(x) when alpha = 2 T\_min = 1')+  
 theme(plot.title = ggplot2::element\_text(hjust=0.5))  
print(p1)



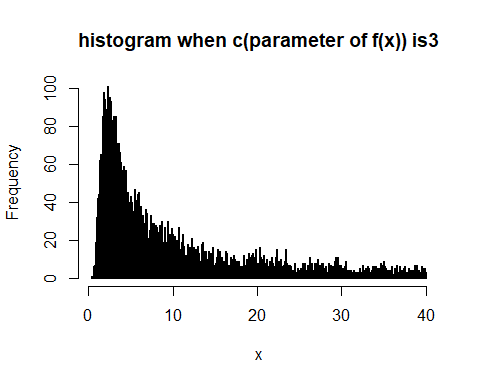
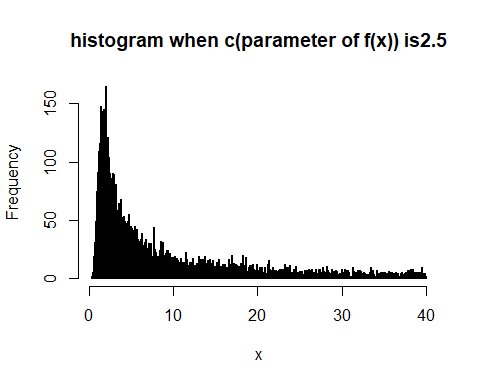
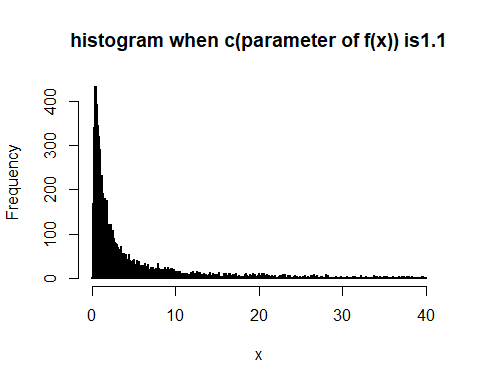
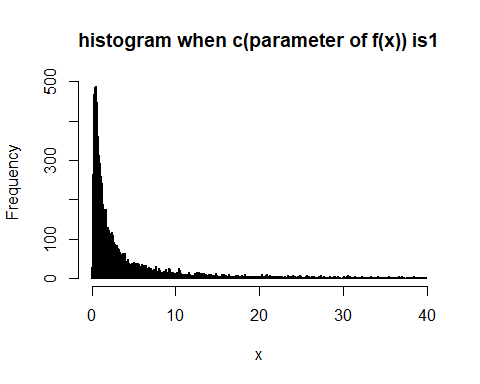
# plot for fmm(x) and f(x)  
c <- 1.1  
t\_min <- 0.45  
alpha <- 1.2  
x\_f <- 1:1000/200  
fx <- c/sqrt(2\*pi)\*exp(-c\*c/2/x\_f)\*x\_f^(-3/2)  
fpx2 <- get\_fpx(alpha,t\_min,x\_f)  
df <- data.frame(x = x\_f,fx =fx,fmmx=fpx2)  
df2 <- melt(df,id.vars='x')  
# plot  
p2 <-ggplot(df2,aes(x=x,y=value))+  
 geom\_point(aes(color=variable))+  
 ggtitle('f(x) and fmm(x) when alpha = 1.2 T\_min = 0.45,c = 1.1')+  
 theme(plot.title = ggplot2::element\_text(hjust=0.5))  
print(p2)



################################################################################  
#q2  
new\_powerLaw\_sampling <- function(nsamples,alpha,t\_min)  
{  
  
 n1 <- nsamples\*(alpha-1)/(alpha)  
 n2 <- nsamples - n1  
 first\_part <- runif(n1,0,t\_min)  
 second\_part <- rplcon(n2,t\_min,alpha)  
 return(c(first\_part,second\_part))  
}  
get\_sample <- function(mc,new\_samples,alpha,t\_min,c\_in\_fx){  
 accept\_sample <- c()  
 for (i in 1:length(new\_samples))   
 {  
 y <- new\_samples[i]  
 u <- runif(1)  
 fxy <- c\_in\_fx/sqrt(2\*pi)\*exp(-c\_in\_fx\*c\_in\_fx/2/y)\*y^(-3/2)  
 if(y <= t\_min)  
 {  
 fyy <- (alpha-1)/t\_min/alpha  
 }else  
 {  
 fyy <- ((alpha-1)/alpha/t\_min)\*(y/t\_min)^-alpha  
 }  
 if(u <= fxy/(fyy\*mc))  
 {accept\_sample <- c(accept\_sample,y)}  
 }  
 return(accept\_sample)  
}  
new\_samples <- new\_powerLaw\_sampling(20000,alpha,t\_min)  
res <- get\_sample(1.8433,new\_samples,alpha,t\_min,c)  
print(1-length(res)/length(new\_samples))

## [1] 0.460023

################################################################################  
#q3  
c\_in\_fx\_list <- c(1,1.1,2.5,3)  
t\_min <- 0.45  
alpha <- 1.2  
reject\_rate <- c()  
mean <- c()  
variance <- c()  
for (i in c\_in\_fx\_list) {  
 # get samples  
 new\_samples <- new\_powerLaw\_sampling(20000,alpha,t\_min)  
 x1 <- 1/(3-2\*alpha)   
 mc1 <- (i/sqrt(2\*pi)\*exp(-i\*i/2/(x1))\*(x1)^(-3/2))/((alpha-1)/t\_min/alpha\*(x1/t\_min)^(-alpha))  
 mc2 <- (i/sqrt(2\*pi)\*exp(-i\*i/2/(1/3))\*(1/3)^(-3/2))/((alpha-1)/t\_min/alpha)  
 res <- get\_sample(max(mc1,mc2,1),new\_samples,alpha,t\_min,i)  
 # get some data  
 reject\_rate <- c(reject\_rate,(length(new\_samples)-length(res))/length(new\_samples))  
 mean <- c(mean,mean(res))  
 variance <- c(variance,var(res))  
 # plot histogram   
 hist(res[which(res<=40)],xlab = 'x',main = paste0('histogram when c(parameter of f(x)) is',i),breaks = 400)  
}



print(mean)

## [1] 19590.78 18687.93 7492441.69 64849.23

print(variance)

## [1] 1.160088e+12 6.324479e+11 5.380586e+17 7.695385e+12

print(reject\_rate)

## [1] 0.4389219 0.4508225 0.4261713 0.4428221