setwd("Z:/732A38/13/labs/3")

population<-read.csv2("population.csv")

#assignment 1

setwd("Z:/732A38/13/labs/3")

population<-read.csv2("population.csv")

#assignment 1

randcities<-function(sample)

{

total<-sum(sample$Population)

Cities<-"";

rand<-runif(1); #This row can be exchanged with "rand<-randNumGen(1)" see the next function

count<-0; #counts where on the 0 to 1 scale the loop is

city<-0; #counts which city we are in

while (rand>count){

city<-city+1;

count<-count+sample$Population[city]/total;

}

Cities<-as.character(sample$Municipality[city])

pop<-as.numeric(sample$Population[city])

return(list(City=Cities, population=pop, id=city));

}

cities<-c();

popul<-c();

population1=population

for (i in 1:20) {

res<-randcities(population1);

cities=c(cities,res$City);

popul<-c(popul,res$population);

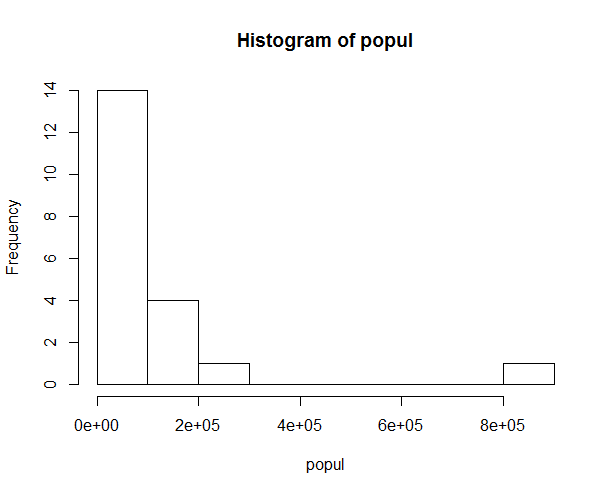
population1=as.data.frame(population1[-res$id,])

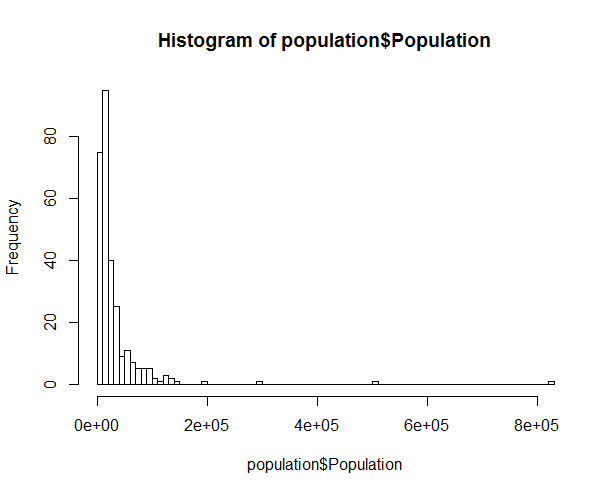
}

res<-randcities(population);

cities

hist(popul);

hist(population$Population,100) 



> cities

[1] "Malmö" "Uddevalla" "Uppsala" "Stockholm" "Bjuv" "Ulricehamn" "Västerås"

[8] "Vänersborg" "Öckerö" "Katrineholm" "Huddinge" "Haninge" "Solna" "Sölvesborg"

[15] "Borås" "Karlstad" "Partille" "Sotenäs" "Örebro" "Kungsbacka"

Large cities are few, but they are present in sample then in the population (because pps-sampling was done)

Assignment 2

**Find cumulative function**

**Find inverse of F(x)**

For x < 0:

For x ≥ 0:

laplace<-function(u){

result<-0;

if(u<0.5){

result<-log(2\*u);

}else {

result<- -log(2\*(1-u));

}

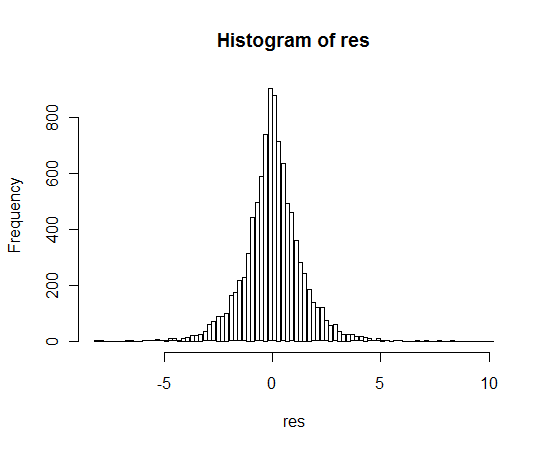
result;

};

l<-runif(10000);

res<-sapply(l, laplace)

hist(res, 100);



acceptance-rejection method

1. Choose a density g(y). In this case, we choose DE(0,1).
2. Find constant c

;

The value for c is obtained as the maximum value of .

x = 1 gives the maximum value of c.

DE(0,1) needs a constant greater than 1 to majorize N(0,1) and c is a measure of the efficiency of the algorithm. It is the best to keep the number of rejection small for maximum efficiency.

1. Generate a random number Y from density g(y)
2. Generate a uniform random number U
3. If

Then accept X = Y

mynorm<-function(n)

{

k<-1;

c<-1.31;

result<-1:n;

rejection <-1:n;

rejection[]<-0;

while(k<n)

{

u1<-runif(1);

y<-laplace(u1);

u<-runif(1);

if(u <=dnorm(y)/(c\*0.5\*exp(-abs(y))) ){

k<-k+1;

result[k]<-y;

} else {

rejection[k]<-rejection[k]+1;

}

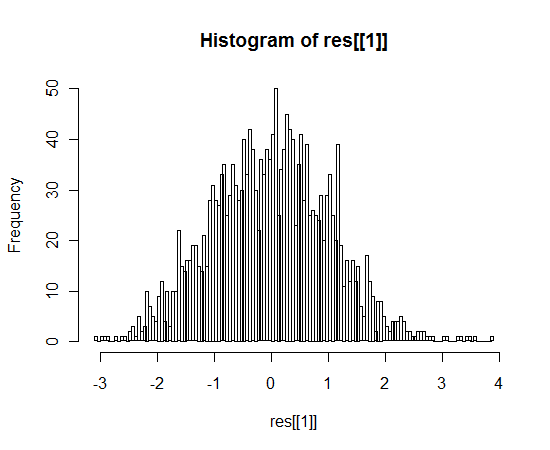
}

list(result,rejection)

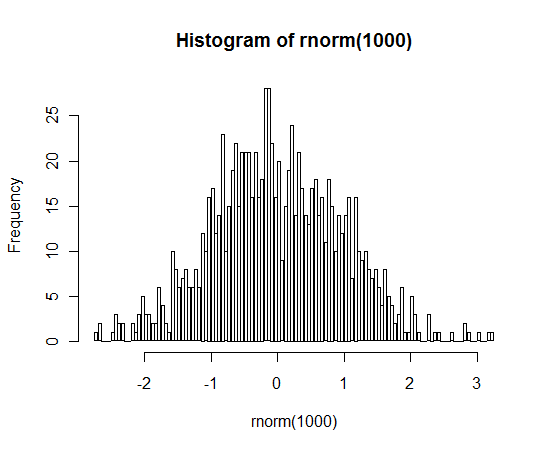
}

res<-mynorm(2000)

hist(res[[1]],100);



hist(rnorm(1000),100);



sum(res[[2]])/(sum(res[[2]])+2000)

[1] 0.2498125

#result is 0.23 which is (c-1)/c

[1] 0.2316558