**Homework 6**

**BIST/STAT 6494 Fall 2017**

**Haim Bar**

**Due Date: October 29, 2017**

**(by 23:59, via HuskyCT)**

**SAS questions:**

1. Count the number of missing values for WBC, RBC and Chol in the Blood data set. Use the MISSING function to detect missing values.

libname hw6 'P:\STAT-6494-Data Management in SAS and R\data';

**data** q1;

set hw6.blood;

if missing(WBC) then MissWBC + **1**;

if missing(RBC) then MissRBC + **1**;

if missing(Chol) then MissChol +**1**;

**run**;

**proc** **print** data=blood(firstobs=**1000**);

var MissChol MissWBC MissRBC;

**run**;

There are 92, 84 and 205 missing values for WBC, RBC and Chol.

1. The SAS data set Psych contains an ID variable, 10 question responses (Ques 1 – Ques 10), and 5 scores (Score1 – Score5). You want to create a new, temporary SAS data set (Evaluate) containing the following:
2. A variable called QuesAve computed as the mean of Ques1 – Ques10. Perform this computation only if there are seven or more non-missing question values.
3. If there are no missing Score values, compute the minimum score (MinScore), the maximum score (MaxScore), and the second highest score (SecondHighest).

**data** Evaluate;

set hw6.Psych;

notmissq=**10**-nmiss(of Ques1-Ques10);

if notmissq ge **7** then QuesAve = mean(of Ques1-Ques10);

misss=nmiss(of Score1-Score5);

if misss = **0** then do;

MinScore=min(of Score1-Score5);

MaxScore=max(of Score1-Score5);

SecondHighest=largest(**2**,of Score1-Score5);

end;

**run**;

1. Write a short DATA \_NULL\_ step to determine the largest integer you can store on your computer in 3, 4, 5, 6, and 7 bytes.

**data** \_NULL\_;

array len{**8**} \_numeric\_ BYTES\_1-BYTES\_8;

do i=**3** to **7**;

len{i} = constant('exactint',i);

end;

put "Decimal:" (BYTES\_3-BYTES\_7) (=/comma32.0);

put"Binary:" (BYTES\_3-BYTES\_7) (=/binary64.);

**run**;

1. Create a temporary SAS data set called Random, consisting of 1,000 observations, each with a random integer from 1 to 5. Make sure that all integers in the range are equally likely. Run PROC FREQ to test this assumption.

**data** Random;

do i= **1** to **1000**;

x=int(ranuni(**0**)\***5**+**1**);

output;

end;

**run**;

**proc** **freq** data=Random;

**run**;

1. Data set Char\_Num contains character variables Age and Weight and numeric variables SS and Zip. Create a new, temporary SAS data set called Convert with new variables NumAge and NumWeight that are numeric values of Age and Weight, respectively, and CharSS and CharZip that are character variables created from SS and Zip. CharSS should contain leading 0s and dashes in the appropriate places for Social Security numbers and CharZip should contain leading 0s.

Hint: The Z5. format includes leading 0s for the ZIP code.

**proc** **format**;

picture ssnpix LOW-HIGH='999-99-9999';

**run**;

**data** Convert;

set hw6.Char\_Num;

NumAge=input(Age,**8.**);

NumWeight=input(Weight,**8.**);

charSS=put(SS,ssn11.);

charZip=put(Zip,Z5.);

drop Age Weight SS Zip;

**run**;

1. Using the Stocks data set, compute daily changes in the prices. Use the statements here to create the plot.

goptions reset = all colors = (black) ftext = swiss htitle = 1.5;

symbol1 v = dot I = smooth;

title “Plot ofDaily Price Differences”;

proc gplot data = difference;

Plot Diff \* Date;

run;

quit;

**data** difference;

set hw6.Stocks;

diff=Dif(Price);

**run**;



**R questions:**

Note: do not use any package in these questions.

1. Simulate 1000 data points from a mixture of two distributions: N(0, 1) and N(5, 2) with mixing parameter 0.3. That means, each data point follows N(0, 1) distribution with probability 0.3, follows N(5, 2) with probability 0.7. Plot a histogram on simulated data with appropriate scale, title and labels.

nrep = 1000 # initialize storage for simulated values

x = rep(0,nrep)

set.seed(2) # set seed to replicate

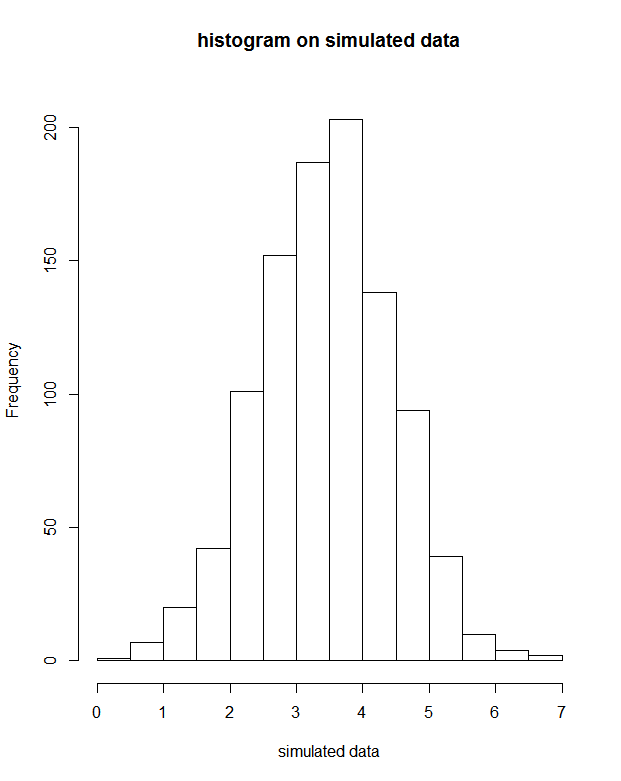
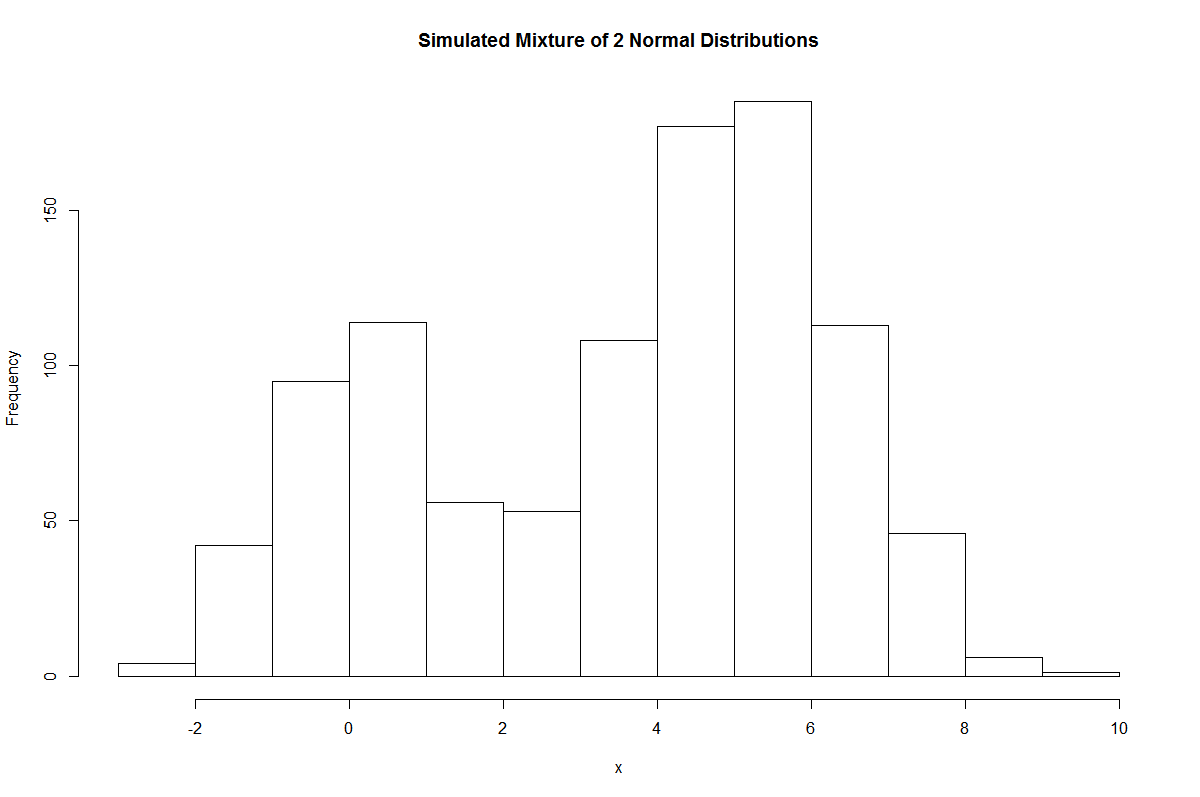
for (i in 1:nrep){

U=runif(1)

if (U <=0.3) x[i] = rnorm(1) else x[i] = rnorm(1, mean = 5, sd = sqrt(2))

}

hist(x,main="Simulated Mixture of 2 Normal Distributions")

1. Simulate 1000 observations of bivariate random variables (xi, yi), i=1,2,…,1000 and create a scatterplot of the simulated data (plot yi by xi).
2. yi = 3xi + , where xi are evenly distributed in (-10, 10). follow a standard normal distribution.

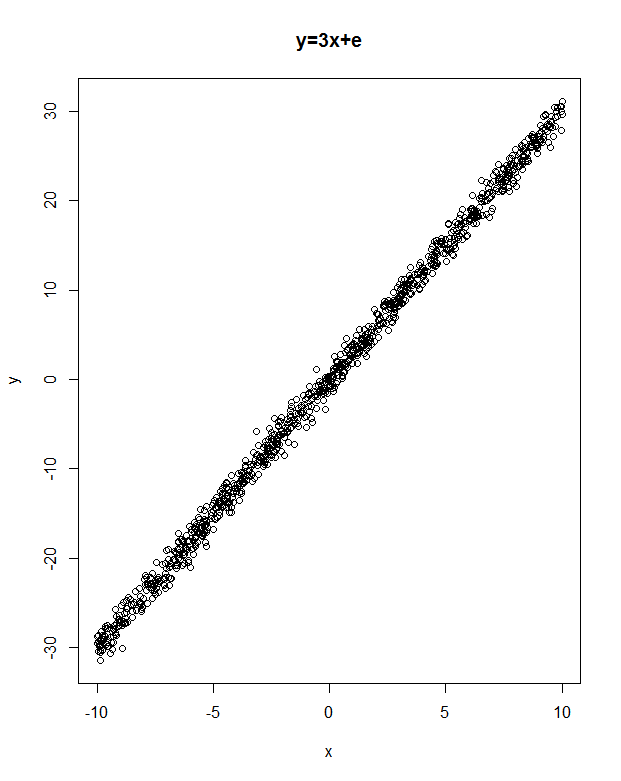
set.seed(750)

x<-runif(1000,-10,10)

e<-rnorm(1000,0,1)

y<-3\*x+e

plot(x,y,main="y=3x+e")



1. (xi, yi) are randomly placed on a circle with radius 10, center (0, 0).

set.seed(750)

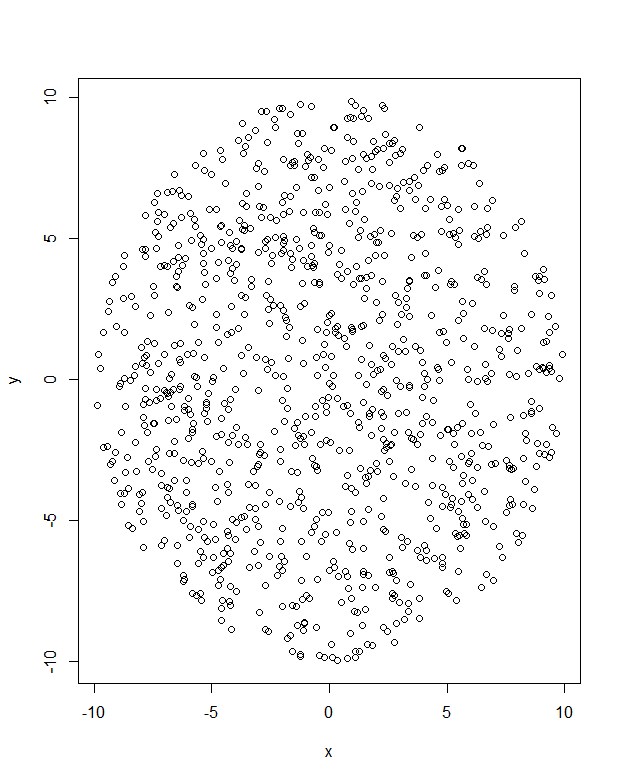
theta<-2\*pi\*runif(1000)

r<-10\*sqrt(runif(1000))

x = r\*cos(theta)

y = r\*sin(theta)

plot(x,y)



1. (xi, yi) is randomly placed in the area of , with the constraint that (xi, yi) cannot be in the circle with radius 1, center(0, 0).

x<-rep(0,1000)

y<-rep(0,1000)

for (i in 1:1000){

a=0

b=0

while(sqrt(a^2+b^2)<=1){

a=runif(1,-10,10)

b=runif(1,-10,10)

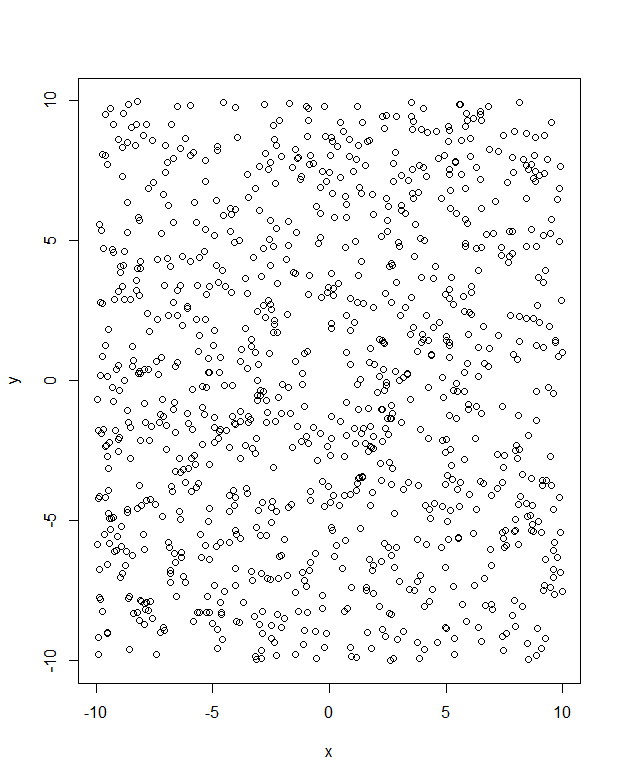
}

x[i] = a

y[i] = b

}

plot(x,y)



1. Define the sequence:

The set A is provided in the file SetA.txt. Write a program which calculates the sequence Do not use loops.

setA<-read.table("P:\\STAT-6494-Data Management in SAS and R\\data\\setA.txt")

A <- setA$x

getSeq <- function(n,a,x){

if(n>1){

x=getSeq(n-1,a,x)

if(n %in% a){

x[n] <- x[n-1] + 1

}else{

x[n] <- 0

}

}

return(x)

}

sequence <- getSeq(100,A,rep(0,100))

sequence

1. The Probability Integral Transformation Theorem states the following: Let *X* have a continuous CDF *F*X, and define a random variable *U* as U=FX(x). Then U is uniformly distributed on [0,1], i.e., P(U≤u)=u for all 0≤u≤1.

This theorem can be used to generate random variables with an arbitrary continuous distribution function *F*, if *F-1* is explicitly known.

In this question you are asked to generate a random sample from an exponential distribution with rate parameter =5. The probability distribution function was provided in the lecture notes.

1. Write the CDF of the exponential distribution, *F*.
2. Obtain the inverse-CDF of the exponential distribution, *F-1*.
3. In R, Generate a random sample of size 50 from a uniform distribution, *ui*, i=1,…,50.

set.seed(88)

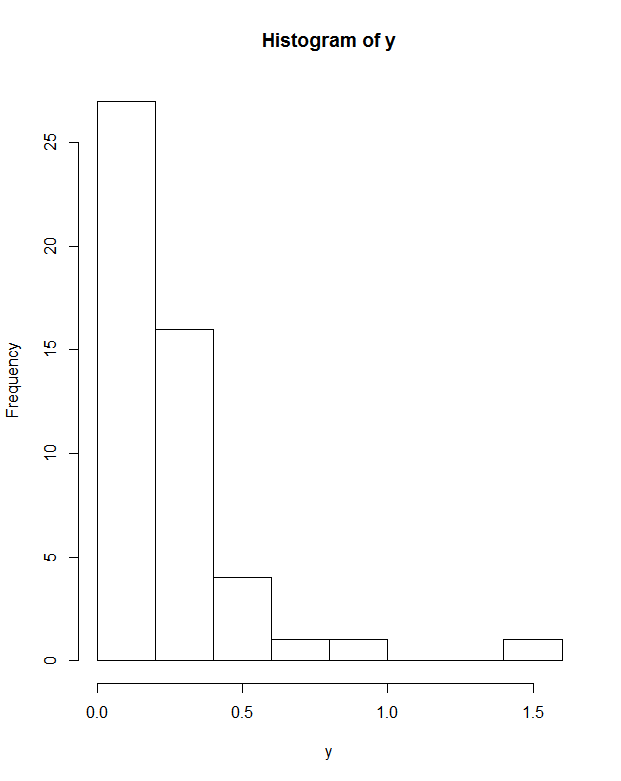
u<-runif(50)

1. In your R program, return the vector *y*=*F-1*(*u*)

y<--log(1-u)/5

1. Plot a histogram of *y*. Does it look like an exponential distribution?

hist(y)



Yes, it does.

1. One possible way to verify that your simulation is correct, is to use the Kolmogorov Smirnov test which is implemented in R (the ks.test function). Run the ks.test function using the vector *y* and specify the correct distribution, Exp(5), and obtain the “p-value” from the test. What is your conclusion based on the p-value?

ks.test(y,pexp,rate=5)$p.value

The p-value is 0.2566>0.05, so accept the null hypothesis that y is from Exp(5).

1. Repeat steps c-f 1000 times, and store the 1000 p-values in a vector called *pvals*.

pvals<-rep(0,1000)

for (i in 1:1000){

u<-runif(50)

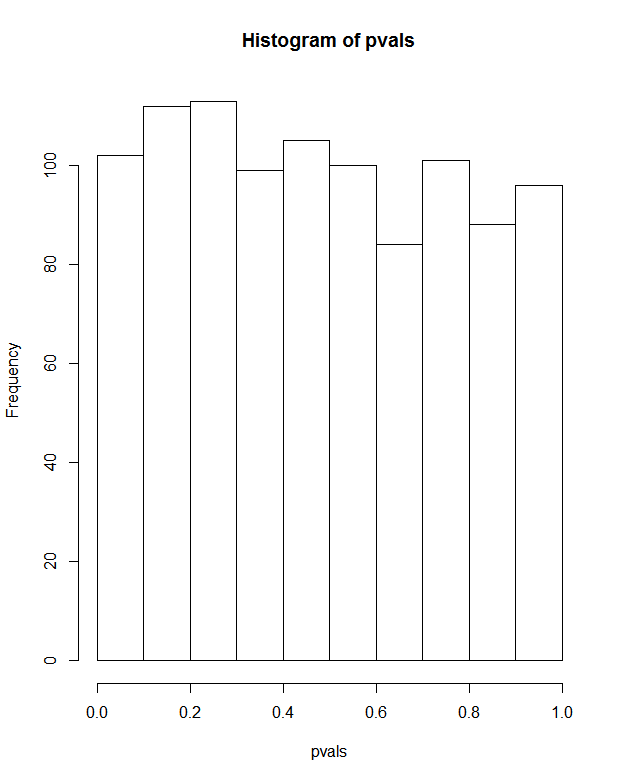
y<--log(1-u)/5

pvals[i]<-ks.test(y,pexp,rate=5)$p.value

}

1. Plot a histogram of *pvals*. What is the distribution of the p-values from the 1000 simulated data sets?

hist(pvals)



It is a uniform distribution.