Homework Assignment 1

Jack, 1930026143

Due on Feb 27, 2022 at 11:59 pm

Instructions: You need to fully show your explanations, codes, and results to get full credit. You will need to submit your R markdown file and the generated pdf file. Missing the R markdown file, you will get a 10% penalty. Missing pdf file, you will have no grades (Your TA will not knit pdf for you). Late submission will not be accepted.

1. a) Generate a random sample $X_1, ., X_{100}$ which is from a normal distribution with mean $\mu = 5$ and standard deviation $\sigma = 3$. Use set.seed(99) before random number generation.

```
set.seed(99)
X = rnorm(100, mean=5, sd=3)
X
```

```
##   [1]  5.64189  6.43897  5.26349  6.33158  3.91149  5.36802  2.40846  6.46887
##   [9]  3.90765  1.11727  2.76269  7.76465  7.25016 -2.52566 -4.12280  5.00080
##  [17]  3.81794 -0.23508  6.49589  5.81286  8.29676  7.25754  4.82175  3.96629
##  [25]  5.66800  6.65536  7.05093  3.36236  0.89769  9.20016  9.11916  6.35077
##  [33]  4.56112  5.38429 -1.88416  0.90029  4.40756  5.20426  5.27151  5.96828
##  [41]  5.39894 -0.03779  4.16456  0.34277  0.86090  0.92883  2.23659  2.39955
##  [49]  9.96993  4.53476  0.27034  6.87225  5.99066  3.80563  1.75581  4.76923
##  [57]  3.42339  6.17387  2.95864  2.75391  4.51125  4.65735  3.66602  5.77088
##  [65]  1.69959  0.99027  5.61470  5.10586  3.68326  6.20122  6.75551  6.44639
##  [73]  2.58550  5.90923  2.52147  5.42429  7.22107  9.15514  3.09927  5.71577
##  [81]  5.10983  5.86558  1.68209  7.67672  8.54441  6.05350  5.18902  9.18279
##  [89]  4.96248  7.03423  4.83156  5.40798 10.59373  8.44613  2.18004  4.81434
##  [97]  3.87173  5.62907  7.11638  6.95627
```

b) Write an R function 'fx' in R to implement the function $y=(x-a)/b$, which will transform an input v

```
fx <- function(x, a, b){
  y = (x-a)/b
  return(y)
}

# test the function 'fx'
# take the three arguments x=10, a=2, b=4
# detemine the result whether equal to (10-2)/4
fx(10, 2, 4) == 2
```

```
## [1] TRUE
```

c) Generate the random sample y using the function in b), where x = the random sample generated in a

```
y = fx(X, 5, 3)
y_mean = mean(y) # sample mean
y_sd = sd(y)     # sample standard deviation
```

Because of the property of the normal distribution, it satisfies: $E(aX+b) = aE(X)+b$ and $Var(aX+b) = a^2Var(X)$ ($Sd(aX+b) = a*Sd(X)$). We can calculate the population mean and standard deviation by the functions.

```
# Population
population_mean <- (5-5)/3  # Population mean
population_sd <- 3/3        # Population standard deviation
```

The results show as follow:

```
print(y_mean) # sample mean
```

```
## [1] -0.104
```

```
print(y_sd)   # sample standard deviation
```

```
## [1] 0.9007
```

```
print(population_mean) # sample mean
```

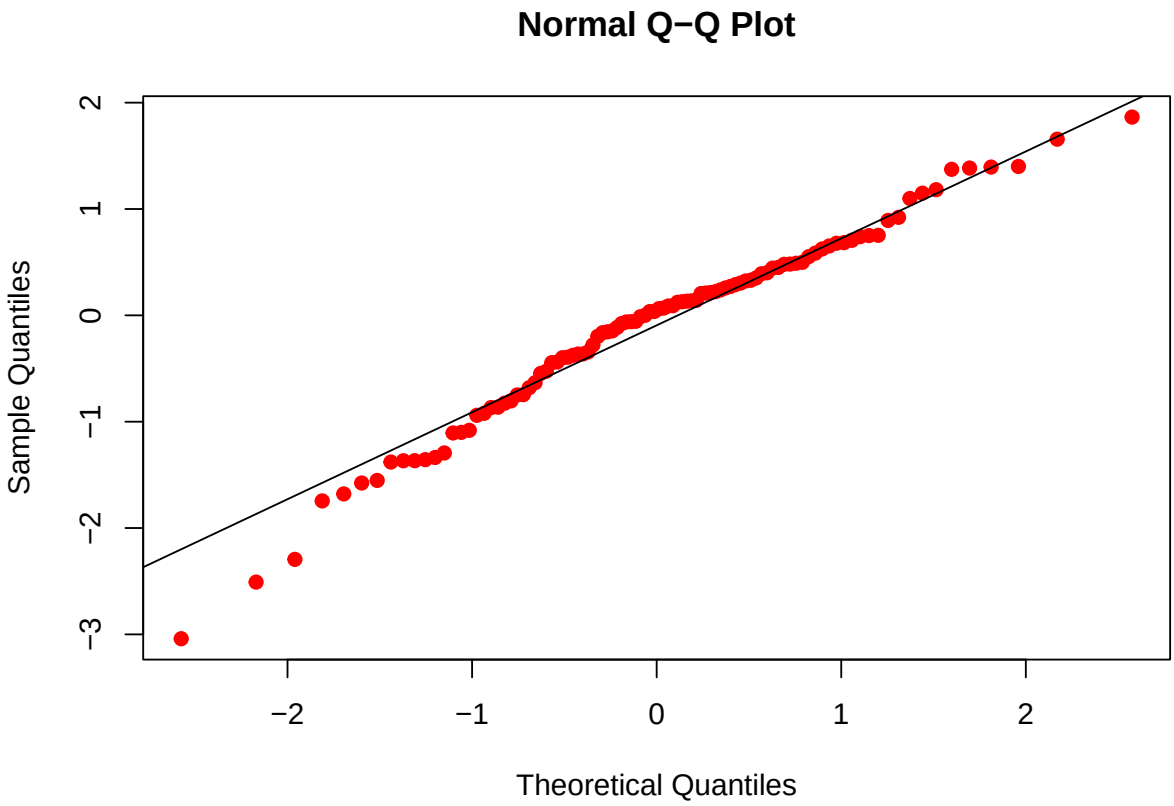
```
## [1] 0
```

```
print(population_sd)   # sample standard deviation
```

```
## [1] 1
```

The distribution of y is normal distribution. Because the population of data x is normal distribution, and the data y is a linear transformation of x. Therefore, y is also satisfies normal distribution. We can also prove by the QQ plot:

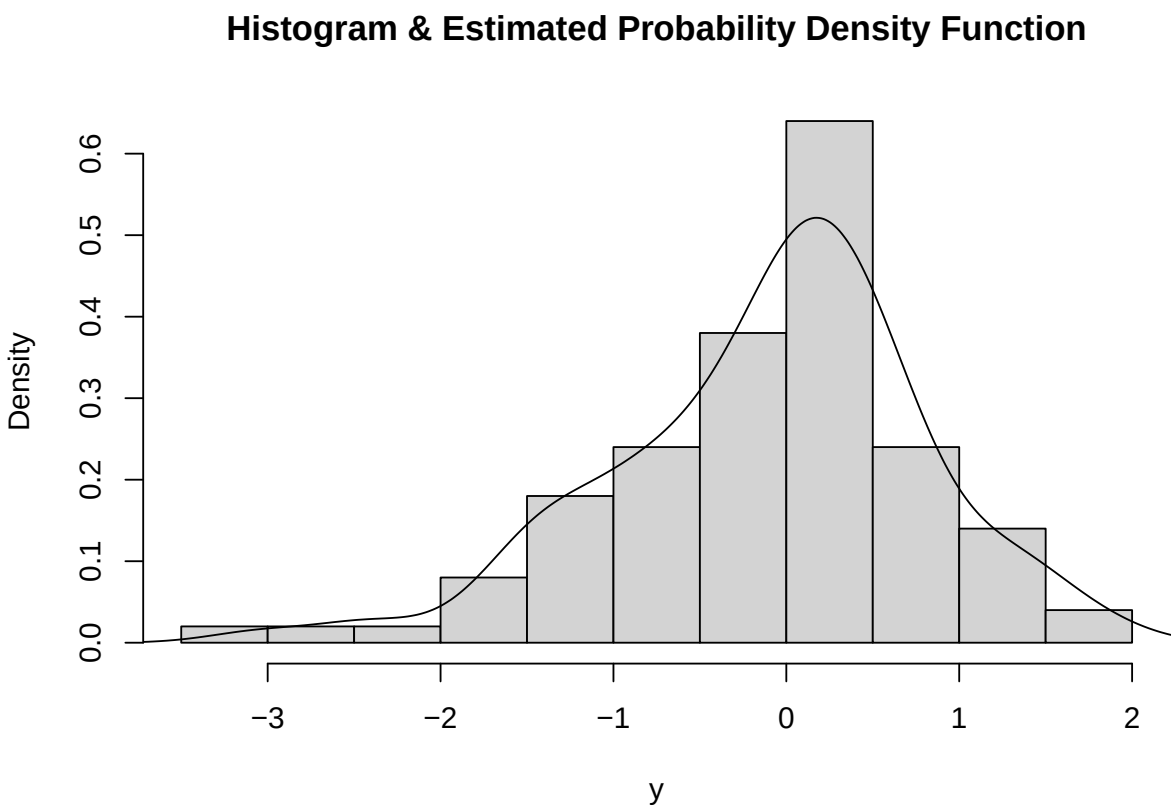
```
qqnorm(y,col="red",pch=19)
qqline(y,col="black")
```



QQ Plot can tell whether a piece of data fits the normal distribution. We can be found that mostly all the points of the plot are around the QQ line, which means that there is sufficient evidence that y also conforms to a normal distribution.

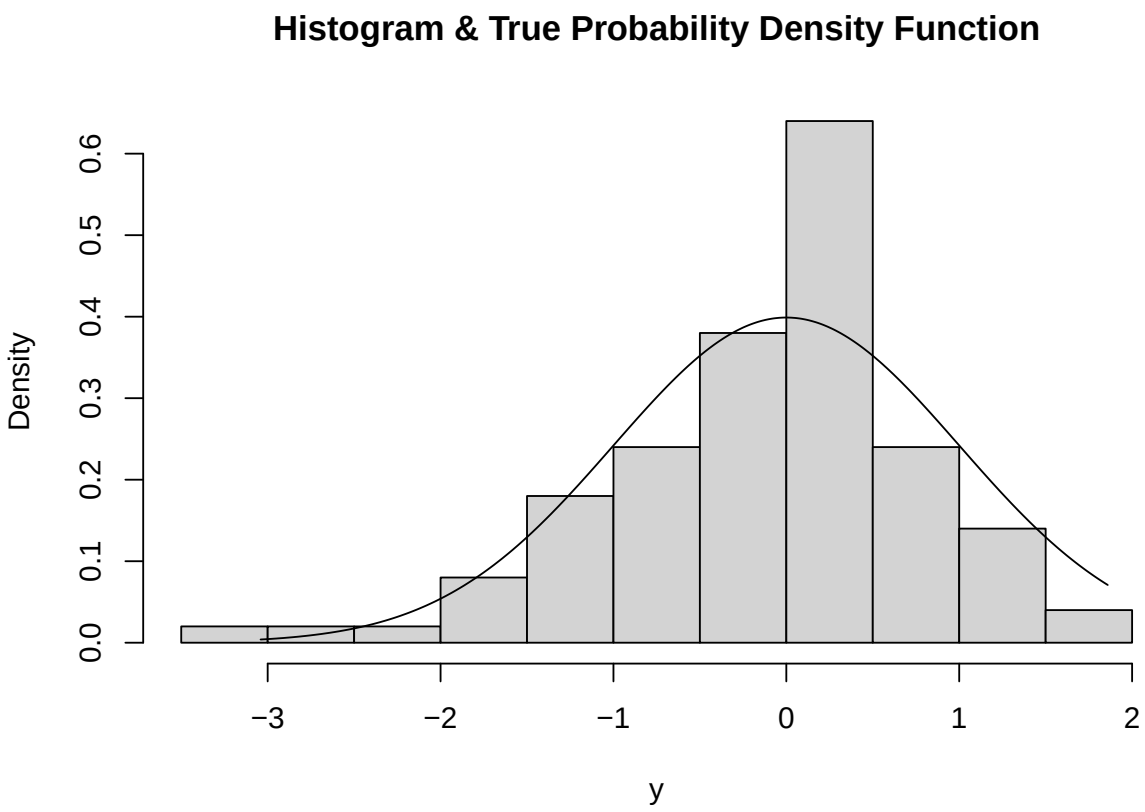
d) Display a probability histogram of the random sample y and add an estimated probability density fu

```
hist(y,main="Histogram & Estimated Probability Density Function",freq=FALSE)
lines(density(y))
```



e) Add the true probability density function to your histogram in d)

```
hist(y,main = "Histogram & True Probability Density Function",freq=FALSE)
d <- seq(from=min(y), to=max(y), by=0.01)
lines(d, dnorm(d,0,1))
```



2. We will use the dataset called `hflights`. This dataset contains all flights departing from Houston airports IAH (George Bush Intercontinental) and HOU (Houston Hobby). The data comes from the Research and Innovation Technology Administration at the Bureau of Transportation statistics: hflights. Make sure you have installed the packages `hflights` before suing them.

```
# Load packages
# install.packages("hflights")
library(hflights)
data(hflights)
library('dplyr');
```

```
##
##     'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

(a) How many rows and columns of `hflights`? Get the names of the columns.

```
r = nrow(hflights)      # return the row numbers of hflights
c = ncol(hflights)      # return the column numbers of hflights
col_name = colnames(hflights)    # return the column names
```

The result shows as follows:

```
print(r)
```

```
## [1] 227496
```

```
print(c)
```

```
## [1] 21
```

```
print(col_name)
```

```
##  [1] "Year"              "Month"             "DayofMonth"
##  [4] "DayOfWeek"         "DepTime"           "ArrTime"
##  [7] "UniqueCarrier"     "FlightNum"         "TailNum"
## [10] "ActualElapsedTime" "AirTime"           "ArrDelay"
## [13] "DepDelay"          "Origin"            "Dest"
## [16] "Distance"          "TaxiIn"            "TaxiOut"
## [19] "Cancelled"         "CancellationCode"  "Diverted"
```

(b) Select the first 15 rows make it a data frame called `phflights`. Suppose we would like check three variables, DepTime, ArrTime and FlightNum. Select these three columns and call it `sflights`. Only Show the first few lines of `sflights`.

phflights: The first 15 rows of hflights

```
phflights <- hflights[1:15,]
head(phflights)
```

```
##      Year Month DayofMonth DayOfWeek DepTime ArrTime UniqueCarrier FlightNum
## 5424 2011     1          1         6    1400    1500            AA       428
## 5425 2011     1          2         7    1401    1501            AA       428
## 5426 2011     1          3         1    1352    1502            AA       428
## 5427 2011     1          4         2    1403    1513            AA       428
## 5428 2011     1          5         3    1405    1507            AA       428
## 5429 2011     1          6         4    1359    1503            AA       428
##      TailNum ActualElapsedTime AirTime ArrDelay DepDelay Origin Dest Distance
## 5424  N576AA                60      40      -10        0    IAH  DFW      224
## 5425  N557AA                60      45       -9        1    IAH  DFW      224
## 5426  N541AA                70      48       -8       -8    IAH  DFW      224
## 5427  N403AA                70      39        3        3    IAH  DFW      224
## 5428  N492AA                62      44       -3        5    IAH  DFW      224
## 5429  N262AA                64      45       -7       -1    IAH  DFW      224
##      TaxiIn TaxiOut Cancelled CancellationCode Diverted
## 5424      7      13         0                         0
## 5425      6       9         0                         0
```

```
## 5426      5      17         0                         0
## 5427      9      22         0                         0
## 5428      9       9         0                         0
## 5429      6      13         0                         0
```

sflights: Three variables, DepTime, ArrTime and FlightNum of hflights

```
sflights <- hflights[ ,colnames(hflights) %in% c('DepTime', 'ArrTime', 'FlightNum')]
head(sflights)
```

```
##      DepTime ArrTime FlightNum
## 5424    1400    1500       428
## 5425    1401    1501       428
## 5426    1352    1502       428
## 5427    1403    1513       428
## 5428    1405    1507       428
## 5429    1359    1503       428
```

(c) Create a new column vector Called `BNum` indicating if the FlightNum is greater than 1000 and append this column to `sflights`. Show the first few lines.

BNum: FlightNum is greater than 1000, and the new column of the sflights

```
BNum = hflights$FlightNum > 1000
sflights$BNum<-BNum
head(sflights)
```

```
##      DepTime ArrTime FlightNum  BNum
## 5424    1400    1500       428 FALSE
## 5425    1401    1501       428 FALSE
## 5426    1352    1502       428 FALSE
## 5427    1403    1513       428 FALSE
## 5428    1405    1507       428 FALSE
## 5429    1359    1503       428 FALSE
```

(d) Compute the average arrival delay (ArrDelay) to each destination for `hflights`. (Hint: use na.rm = TRUE to remove missing values) Only show the first 10 results. Then for each carrier, calculate the percentage of flights cancelled or diverted.

The average arrival delay (ArrDelay) to each destination for hflights:

```
dest_arr <- group_by(hflights, Dest)
dest_AvgDelay <- summarise(dest_arr, ArrDelay=mean(ArrDelay, na.rm=TRUE))
dest_AvgDelay[1:10,]
```

```
## # A tibble: 10 x 2
##    Dest  ArrDelay
##    <chr>    <dbl>
##  1 ABQ       7.23
##  2 AEX       5.84
##  3 AGS       4
```

```
##  4 AMA       6.84
##  5 ANC      26.1
##  6 ASE       6.79
##  7 ATL       8.23
##  8 AUS       7.45
##  9 AVL       9.97
## 10 BFL     -13.2
```

The percentage of flights cancelled or diverted for each carrier:

```
    carr_fli <- group_by(hflights, UniqueCarrier)
    cancel_devert <- summarise(carr_fli , Cancelled=mean(Cancelled), Diverted=mean(Diverted))
    head(cancel_devert)
```

```
## # A tibble: 6 x 3
##   UniqueCarrier Cancelled Diverted
##   <chr>             <dbl>    <dbl>
## 1 AA              0.0185   0.00185
## 2 AS              0        0.00274
## 3 B6              0.0259   0.00576
## 4 CO              0.00678  0.00263
## 5 DL              0.0159   0.00303
## 6 EV              0.0345   0.00318
```