STOR 565 Fall 2019 Homework 1

Due on 09/03/2019 in Class

Hunter Finger

Exercise 1. *(5 pt)* Using the `c`, `rep` or `seq` commands, create the following 6 vectors:

```
x1 = c(2, .5, 4, 2)
x2 = c(x1,  seq(from = 1, to = 1, length.out = 4))
x3 = seq(from = 1, to = -2, length.out = 4)
x4 = c("Hello", "", "World", "!", "Hello World!")
x5 = c(T, T, NA, F)
x6 = c(1,2,1,2,1,1,2,2)
```

Exercise 2. *(5 pt)* Using `matrix`, and `rbind`, create

$$\mathbf{X} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 0 & -1 & -2 \\ 2 & .5 & 4 & 2 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

More precisely first define a set of four vectors corresponding to the rows of the above matrix and then use rbind to make a corresponding matrix. Note: you will need to play around with the `deparse.level` option in `rbind` to get the matrix as above.

```
a = seq(1, 4, 1)
b = seq(1, -2, -1)
c = c(2, .5, 4, 2)
d = rep(1, 4)
X <- rbind(a, b, c, d, deparse.level = 0)
X
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1  2.0    3    4
## [2,]    1  0.0   -1   -2
## [3,]    2  0.5    4    2
## [4,]    1  1.0    1    1
```

Exercise 3. *(4 pt)*: Consider the matrix X from Exercise 2.

- Make a new vector y1 consisting of all the elements of X which are negative (strictly less than zero).

```
y1 <- X[X < 0]
```

- Make a new vector y2 consisting of all the elements of X which are at strictly positive but less than 2.

```
y2 <- X[X > 0 & X < 2]
```

Exercise 4. *(5 pt)* Applying the conditional selection technique (see the section "indexing" and do not use *subset*), extract the record of student 003 i.e their id number, and their scores in the two tests.

```
students <- data.frame( id      = c("001", "002", "003"), # ids are characters
                        score_A = c(95, 97, 90),          # scores are numericss
                        score_B = c(80, 75, 84))
students
```

```r
stud_record = students[3,]
stud_record
```

```
##    id score_A score_B
## 3 003      90      84
```

Exercise 5. *(10 pt)* Create a data.frame object to display the calendar for Jan 2018 as follows.

```r
## Sun Mon Tue Wed Thu Fri Sat
##      NY   2   3   4   5   6
##   7   8   9  10  11  12  13
##  14 MLK  16  17  18  19  20
##  21  22  23  24  25  26  27
##  28  29  30  31
weekdays =  c("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat")
week1 = c("", "NY", seq(2,6,1))
week2 = seq(7,13,1)
week3 = c(14, "MLK", seq(16,20,1))
week4 = seq(21,27,1)
week5 = c(seq(28,31,1), rep("", 3))

days = rbind(week1, week2, week3, week4, week5)
days = data.frame(days)
colnames(days) = c(weekdays)

print(days, row.names = F)
```

```
##  Sun Mon Tue Wed Thu Fri Sat
##       NY   2   3   4   5   6
##    7   8   9  10  11  12  13
##   14 MLK  16  17  18  19  20
##   21  22  23  24  25  26  27
##   28  29  30  31
```

Exercise 6. *(5 pt)* Create a factor variable `grade` in `students3`, where the `score` variable is divided into $[90, 100]$, $[80, 90)$ and $[0, 80)$ corresponding to A, B and C in `grade` respectively.

```r
id        <- rep(c("001","002","003"), 2)
subj      <- rep(c("A","B"), each = 3)
score     <- c(95, 97, 90, 80, 75, 84)
students3 <- data.frame(id, subj, score)  # try cbind(id, subj, score) to see the difference

# students3$id and students3$subj are automatically formatted as factors
class(students3$id)
levels(students3$id)

class(students3$subj)
levels(students3$subj)

# combind student 003 with 002 via level rename
students4 <- students3    # work on a copy in case of direct modification of students3
levels(students4$id)[3] <- "002"
levels(students4$id)
students4
```

Hint. Functions `cut` to obtain the grades and `transform` to obtain the students5 from stuents3.

```r
students5 <- transform(students3, grade = (cut(score, breaks = c(0, 80, 90, Inf), labels = c('C', 'B',

students5
```

```
##    id subj score grade
## 1 001    A    95     A
## 2 002    A    97     A
## 3 003    A    90     B
## 4 001    B    80     C
## 5 002    B    75     C
## 6 003    B    84     B
```

Exercise 7. *(10 pt)* Without using the `var` and `scale` functions, compute the sample mean and sample covariance `X.var` of the data matrix `X` as in **Exercise 2.** More precisely, think of the i-th row of the matrix as observation of features for i-th individual.

a Create a 4-dimensional vector called `mu` where the i-th row is the mean of the i-th column of X.

```r
mu = array(data = NA, dim = c(4,1,1))
for(i in 1:4){
    mu[i] = mean(X[,i])
}
mu
```

```
## , , 1
##
##       [,1]
## [1,] 1.250
## [2,] 0.875
## [3,] 1.750
## [4,] 1.250
```

b Create a four-dimensional matrix `X.var`

$$X.var = \frac{1}{3}\sum_{i=1}^{4}(\mathbf{x}_{i\cdot} - \mu)(\mathbf{x}_{i\cdot} - \mu)^T$$

where $\mathbf{x}_{i\cdot}$ is the i-th row.

```r
X.var = matrix(NA, 4, 4)

for(i in 1:4){
  for(j in 1:4){
    X.var[i,j] = sum((X[i,j] - mu[i]) * t(X[i,j] - mu[i]))/3
  }
}
X.var
```

Exercise 8. *(10 pt)* Imagine that we wanted to make students aware for each of their subjects, the average score of all other students in that subject. Create a variable (or column) called `score.mean` in `students3`, where next to each student and subject, the value of the score.mean is the average value of all students taking that subject.

```r
students3meanA = mean(students3$score[1:3])
students3meanB = mean(students3$score[4:6])
score.mean = c(rep(students3meanA, 3), rep(students3meanB,3))
students3$score.mean = score.mean
students3
```

```
##    id subj score score.mean
## 1 001    A    95   94.00000
## 2 002    A    97   94.00000
## 3 003    A    90   94.00000
## 4 001    B    80   79.66667
## 5 002    B    75   79.66667
## 6 003    B    84   79.66667
```

Exercise 9. *(15 pt)* Write a function `bisect(f, lower, upper, tol = 1e-6)` to find the root of the univariate function `f` on the interval [`lower`, `upper`] with precision tolerance $\leqslant$ `tol` (defaulted to be 10^{-6}) via bisection, which returns a list consisting of `root`, `f.root` (`f` evaluated at `root`), `iter` (number of iterations) and `estim.prec` (estimated precision). Apply it to the function

$$f(x) = x^3 - 2x - 1$$

on $[1, 2]$ with precision tolerance 10^{-6}. Compare it with the built-in function `uniroot`.

```r
bisec.func = function(f, lower, upper, tol = 1e-6){

  N = 1
  while(N < 1000){
  c = ((lower + upper)/2)
  if(f(c) == 0 | (upper - lower) / 2 < tol){
    iter = N
    root = c
    f.root = f(c)
    estim.prec = (c - lower)
    return_vec = rep(NA, 4)
    return_vec = c(iter, root, f.root, estim.prec)
    return(return_vec)
  } else {
    N = N + 1
    if(sign(f(c)) == sign(f(lower))){
      lower = c
    } else {
      upper = c
    }

  }
  }
}

f = function(x){
    x^3 - 2*x - 1
}
upper = 2
lower = 1

bisec.func(f, lower, upper)
```

```
## [1]  2.000000e+01  1.618033e+00 -3.393219e-06  9.536743e-07
```

```r
uniroot(f, 1:2)
```

```
## $root
## [1] 1.618036
##
```

```
## $f.root
## [1] 9.230512e-06
##
## $iter
## [1] 6
##
## $init.it
## [1] NA
##
## $estim.prec
## [1] 6.103516e-05
```

Exercise 10 *(16 pt)* In the folder for HW 1, you can find data on UNC salaries as a unc_salary_data.csv file (all of which are publicly available and scraped by Ryan Thornburg).

a Read the data using read.csv into a data frame called `salaries`

```
salaries = read.csv("unc_salary_data.csv")
str(salaries)
```

```
## 'data.frame':    12287 obs. of  14 variables:
##  $ name    : Factor w/ 12270 levels "AARON, NANCY G",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ campus  : Factor w/ 1 level "UNC-CH": 1 1 1 1 1 1 1 1 1 1 ...
##  $ dept    : Factor w/ 304 levels "Acad Sup Prog Student-Athletes",..: 234 163 160 175 238 175 55 71
##  $ position: Factor w/ 4120 levels ".Net Developer/Programmer",..: 3597 634 3474 41 3836 2282 41 338
##  $ exempt2 : Factor w/ 3 levels "Exempt","Non-permanent",..: 1 1 3 3 3 3 3 1 1 1 ...
##  $ employed: int  9 9 12 12 12 12 12 12 12 12 ...
##  $ hiredate: int  20030701 19990101 20110912 20090420 20120103 20051003 19960923 20130401 19870101 2
##  $ fte     : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ status  : Factor w/ 5 levels "Continuing","Fixed-Term",..: 2 1 3 3 3 3 3 1 1 1 ...
##  $ stservyr: int  11 17 3 5 2 15 34 11 27 2 ...
##  $ statesal: int  46350 173000 0 0 41696 56588 41707 0 0 0 ...
##  $ nonstsal: int  0 0 38170 50070 0 4412 0 80227 55803 32889 ...
##  $ totalsal: int  46350 173000 38170 50070 41696 61000 41707 80227 55803 32889 ...
##  $ age     : int  55 57 54 29 35 41 62 36 64 26 ...
```

```
head(salaries)
```

```
##                          name campus                           dept
## 1              AARON, NANCY G UNC-CH              Romance Languages
## 2     ABARBANELL, JEFFERY S UNC-CH Kenan-Flagler Business School
## 3                ABARE, BETSY UNC-CH  Institute of Marine Sciences
## 4              ABATE, AARON B UNC-CH       Medicine Administration
## 5        ABATEMARCO, JODI M UNC-CH            School of Education
## 6 ABBOTT-LUNSFORD, SHELBY L UNC-CH       Medicine Administration
##                      position                          exempt2 employed
## 1              Senior Lecturer                           Exempt        9
## 2          Associate Professor                           Exempt        9
## 3         Research Technician Subject to State Personnel Act       12
## 4       Accounting Technician Subject to State Personnel Act       12
## 5 Student Services Assistant Subject to State Personnel Act       12
## 6               HR Consultant Subject to State Personnel Act       12
##   hiredate fte     status stservyr statesal nonstsal totalsal age
## 1 20030701   1 Fixed-Term       11    46350        0    46350  55
## 2 19990101   1 Continuing       17   173000        0   173000  57
## 3 20110912   1  Permanent        3        0    38170    38170  54
## 4 20090420   1  Permanent        5        0    50070    50070  29
```

```
## 5 20120103   1  Permanent       2    41696       0    41696  35
## 6 20051003   1  Permanent      15    56588    4412    61000  41
```

Use `str(salaries)` and `head(salaries)` to get an idea of the data set.

b Make a new data frame called `relevant` consisting only of the columns: name, dept, age,totalsal. (Hint: consider the `subset` function).

```
relevant = subset(salaries, select = c(name, dept, totalsal, age))
```

c Make a new data frame called `top_200` consisting of the information in `relevant` of faculty who make more than $200,000.

```
top_200 = subset(relevant, totalsal > 200000)
```

d Choose 3 departments that you are interested in. Compute the average salary of faculty in these 3 departments.

```
comp_sci = subset(relevant, dept == "Computer Science")
stats = subset(relevant, dept == "Statistics and Operations Res")
art_hist = subset(relevant, dept == "Art")

comp_avg = mean(comp_sci$totalsal)
stat_avg = mean(stats$totalsal)
art_avg = mean(art_hist$totalsal)

avg_sal = c(comp_avg, stat_avg, art_avg)
avg_sal
```

```
## [1] 103307.32 100470.83  77738.34
```

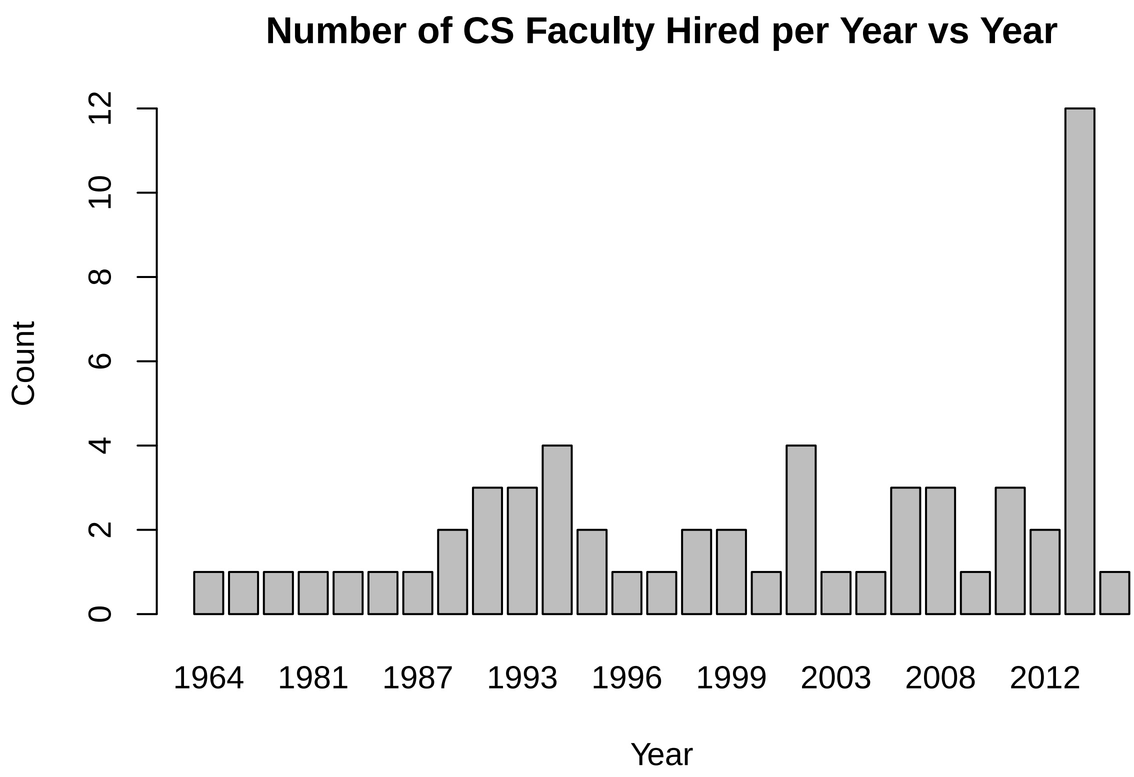
Exercise 11. *(10 pt)* `iris` is a built-in dataset in **R**. Check `?iris` for more information. This dataset has data on 50 flowers each from 3 species of Iris (setosa, versicolor, and virginica). Randomly divide `iris` into five subsets `iris1` to `iris5` (without replacement), thus each subset has 30 rows of the iris data and further stratified to `iris$Species` (namely every subset should have 10 rows from each of the 3 species).

```
iris_df = data.frame(iris)
iris1 = iris_df[sample(nrow(iris_df), 30, replace = F),]
iris2 = iris_df[sample(nrow(iris_df), 30, replace = F),]
iris3 = iris_df[sample(nrow(iris_df), 30, replace = F),]
iris4 = iris_df[sample(nrow(iris_df), 30, replace = F),]
iris5 = iris_df[sample(nrow(iris_df), 30, replace = F),]
iris.5fold <- list(iris1, iris2, iris3, iris4, iris5)
```

Exercise 12 *(10 pt)*

a Recall the UNC salary data set. From the `salaries` data frame plot the number of CS faculty hired per year vs year.

```
salaries$hireyear = floor(salaries$hiredate / 10000)
comp_sci_plot = subset(salaries, dept == "Computer Science")
barplot(table(comp_sci_plot$hireyear), main = "Number of CS Faculty Hired per Year vs Year", ylab = "Co
```



b Now add STOR, Math and Physics to the above plot

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

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

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

```r
all_plot=subset(salaries, dept == "Computer Science"| dept == "Mathematics" | dept == "Statistics and Op
```

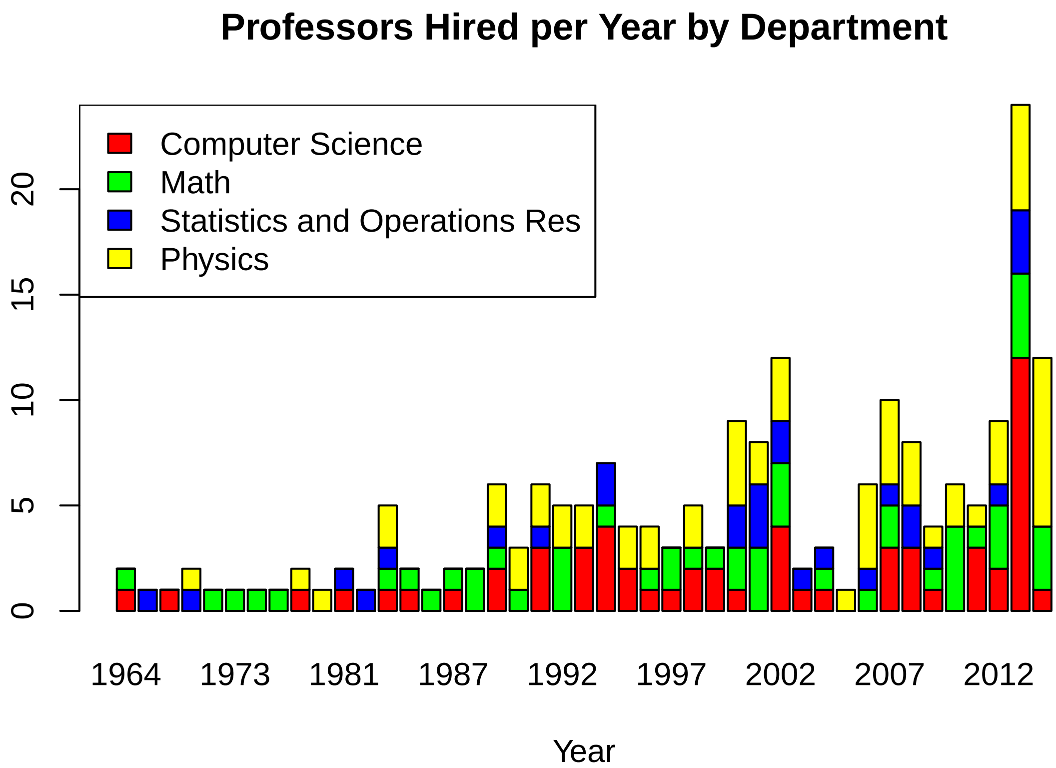
```
## Warning: The `printer` argument is deprecated as of rlang 0.3.0.
## This warning is displayed once per session.
```

```r
a = matrix(NA, 4, 43)


c = c(1,0,1,0,0,0,0,0,1,0,1,0,1,1,0,1,0,2,0,3,0,3,4,2,1,1,2,2,1,0,4,1,1,0,0,3,3,1,0,3,2,12,1)
m= c(1,0,0,0,1,1,1,1,0,0,0,0,1,1,1,1,2,1,1,0,3,0,1,0,1,2,1,1,2,3,3,0,1,0,1,2,0,1,4,1,3,4,3)
s= c(0,1,0,1,0,0,0,0,0,0,1,1,1,0,0,0,0,1,0,1,0,0,2,0,0,0,0,0,2,3,2,1,1,0,1,1,2,1,0,0,1,3,0)
p= c(0,0,0,1,0,0,0,0,1,1,0,0,2,0,0,0,0,2,2,2,2,2,0,2,2,0,2,0,4,2,3,0,0,1,4,4,3,1,2,1,3,5,8)

a = rbind(c,m,s,p)
rownames(a) = c("Computer Science", "Mathematics", "Statistics and Operations Res", "Physics-Astronomy"
colnames(a) = unique(all_plot$hireyear)
barplot(a,
main = "Professors Hired per Year by Department",
```

```
xlab = "Year",
col = c("red","green", "blue", "yellow")
)
legend("topleft",
c("Computer Science","Math", "Statistics and Operations Res", "Physics"),
fill = c("red","green", "blue","yellow"))
```



Exercise 13. *(15 pt)* The following code generates the ensuing plot about `Sepal.Length` in `iris`.

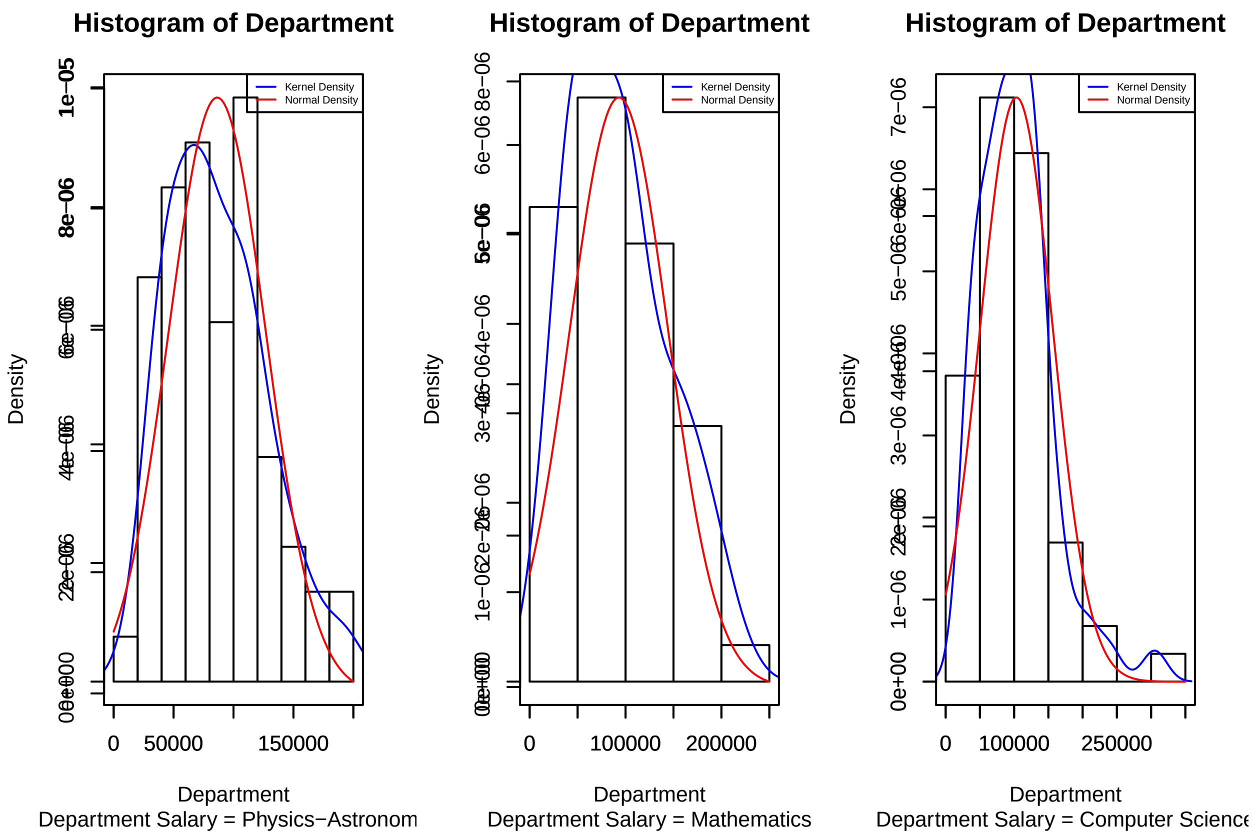
```
e13 = subset(salaries, dept == "Computer Science"| dept == "Mathematics" | dept =="Physics-Astronomy")
opar <- par(mfrow = c(1,3))
  for(l in unique(droplevels(e13$dept)))
  {
   Department <- subset(e13, dept == l, select = totalsal)[[1]]
   h <- hist(Department, sub = paste("Department Salary =", l), freq = FALSE)

   par(new = TRUE)    # add to the current plot
   # Empirical density curve
   lines(density(Department),
        xlim = range(h$breaks),  ylim = c(min(Department), max(Department)),# to match the plotting ran
        col = "blue",
        main = "", sub = "", xlab = "", ylab = ""     # to supress labels
        )
   par(new = TRUE)    # add to the current plot

   # Normal density curve
   curve(dnorm(x, mean = mean(Department), sd = sd(Department)),
         xlim = range(h$breaks),# to match the plotting range
         col = "red",
         main = "", sub = "", xlab = "", ylab = ""     # to supress labels
```

```r
        )

  legend("topright",
         legend = c("Kernel Density", "Normal Density"),
         col = c("blue", "red"), lty = 1, cex = 0.5)
}
```




```r
par(opar)
```