STOR 565 Fall 2019 Homework 1

Ted Henson

*Remark.* This homework aims to introduce you to the basics in **R**, which would be the main software we shall work on throughout this course. It might look like a long homework but it has only 13 problems. The rest are explanations regarding basic things in R. **Total number of points**: 120. Recall all Homeworks are worth the same weight when I compute your final grade. I will convert each HW to out of 100 then and compute the average of your HW scores.

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

x1 = (2, .5, 4, 2);

x2 = (2, .5, 4, 2, 1, 1, 1, 1);

x3 = (1, 0, -1, -2);

x4 = (“Hello”," “,”World“,”!“,”Hello World!");

*Note:* The quotation marks and sometimes the exclamations marks are rendered a little funky in the pdf/html. Just go with it.

**Hint.** For x4, take this opportunity to experiment with the paste function.

x5 = (TRUE, TRUE, NA, FALSE);

**Remark.** Check ?NA and class(NA) to learn more about the missing value object NA. This is not relevant for x5.

x6 = (1, 2, 1, 2, 1, 1, 2, 2).

library(tinytex)

## Warning: package 'tinytex' was built under R version 3.5.3

x1 = c(2, .5, 4, 2)  
x1

## [1] 2.0 0.5 4.0 2.0

x2 = c(2, .5, 4, 2, 1, 1, 1, 1)  
x2

## [1] 2.0 0.5 4.0 2.0 1.0 1.0 1.0 1.0

x3 = c(1, 0, -1, -2)  
x3

## [1] 1 0 -1 -2

x4 = paste("Hello"," ","World", "!", "Hello World!", sep = "")  
x4

## [1] "Hello World!Hello World!"

x5 = c(rep(TRUE, 2), NA, FALSE)  
x5

## [1] TRUE TRUE NA FALSE

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

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.

r1 = c(1, 2, 3, 4)  
r2 = c(1, 0, -1, -2)  
r3 = c(2, .5, 4, 2)  
r4 = (rep(1, 4))  
X = rbind(r1, r2, r3, r4)  
X

## [,1] [,2] [,3] [,4]  
## r1 1 2.0 3 4  
## r2 1 0.0 -1 -2  
## r3 2 0.5 4 2  
## r4 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]  
y1

## [1] -1 -2

* 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]  
y2

## [1] 1.0 1.0 1.0 0.5 1.0 1.0 1.0

**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[which(students$id == "003"), ]

## 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.

## 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   
calendar = data.frame(list( Sun = c('', 7, 14, 21, 28),  
 Mon = c('NY', 8, 'MLK', 22, 29),  
 Tue = c(2, 9, 16, 23, 30),  
 Wed = c(3, 10, 17, 24, 31),  
 Thu = c(4, 11, 18, 25, ''),  
 Fri = c(5, 12, 19, 26, ''),  
 Sat = c(6, 13, 20, 27, '')))  
calendar

Ignore the ## symbols this was just so the above acts like a comment in R.

1) The character object "" for the spaces; 2) the option row.names = FALSE in print function.

**Exercise 6.** *(5 pt)* Create a factor variable grade in students3, where the score variable is divided into , and corresponding to A, B and C in grade respectively.

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

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)  
  
library(tidyverse, warn.conflicts = F)

## Warning: package 'tidyverse' was built under R version 3.5.2

## -- Attaching packages ---------------------------------------------------- tidyverse 1.2.1 --

## v ggplot2 3.1.1 v purrr 0.3.2   
## v tibble 2.1.1 v dplyr 0.8.0.1  
## v tidyr 0.8.3 v stringr 1.4.0   
## v readr 1.3.1 v forcats 0.4.0

## Warning: package 'ggplot2' was built under R version 3.5.3

## Warning: package 'tibble' was built under R version 3.5.3

## Warning: package 'tidyr' was built under R version 3.5.2

## Warning: package 'readr' was built under R version 3.5.2

## Warning: package 'purrr' was built under R version 3.5.3

## Warning: package 'dplyr' was built under R version 3.5.2

## Warning: package 'stringr' was built under R version 3.5.2

## Warning: package 'forcats' was built under R version 3.5.2

## -- Conflicts ------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

students5 <- students3 %>% mutate(grade = cut(score, breaks = c(0, 80, 90, 100),  
 include.lowest = T,  
 right = F))  
students5

## id subj score grade  
## 1 001 A 95 [90,100]  
## 2 002 A 97 [90,100]  
## 3 003 A 90 [90,100]  
## 4 001 B 80 [80,90)  
## 5 002 B 75 [0,80)  
## 6 003 B 84 [80,90)

**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 -th row of the matrix as observation of features for -th individual.

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

mu = c(rep(0, 4))  
mu = apply(X, MARGIN = 2, FUN = mean)  
mu

## [1] 1.250 0.875 1.750 1.250

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

where is the -th row.

dif = X - mu  
  
X.Var = (1/3) \* dif \* t(dif)  
X.Var

## [,1] [,2] [,3] [,4]  
## r1 0.02083333 0.0312500 0.14583333 -0.22916667  
## r2 0.03125000 0.2552083 0.78125000 0.23958333  
## r3 0.14583333 0.7812500 1.68750000 -0.02083333  
## r4 -0.22916667 0.2395833 -0.02083333 0.02083333

**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.

students3 = students3 %>% group\_by(subj) %>% mutate(score.mean = mean(score))  
students3

## # A tibble: 6 x 4  
## # Groups: subj [2]  
## id subj score score.mean  
## <fct> <fct> <dbl> <dbl>  
## 1 001 A 95 94   
## 2 002 A 97 94   
## 3 003 A 90 94   
## 4 001 B 80 79.7  
## 5 002 B 75 79.7  
## 6 003 B 84 79.7

**Exercise 9.** *(15 pt)* The bisection method if a root-finding algorithm from numerical analysis to find a root of a continuous function in an interval once you know that function has different signs at the end points of the interval (i.e.  or vice-versa). Read about this in the [Wikipedia link](https://en.wikipedia.org/wiki/Bisection_method).

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 tol (defaulted to be ) 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

on with precision tolerance . Compare it with the built-in function uniroot.

f = function(x){x^3 - 2\*x - 1}  
  
bisect = function(f, lower, upper, tol = 1e-6){  
   
 iter = 1  
 while(iter <= 10000){  
   
 c = (upper + lower) / 2  
   
 f.root = f(x = c)  
 estim.prec = abs(0 - f.root)  
 if( estim.prec <= tol){  
 return( c(root = c, f.root = f.root, iter = iter, estim.prec = estim.prec))  
 }  
   
 else{  
 iter = iter + 1  
   
 if(sign(f(upper)) != sign(f.root)){  
   
 lower = c  
 }  
   
 else{  
 upper = c  
   
 }  
  
 }  
 iter = iter + 1  
 }  
   
}  
  
bisect(f, lower = 1, upper = 2)

## root f.root iter estim.prec   
## 1.618034e+00 -6.017670e-07 4.100000e+01 6.017670e-07

uniroot(f, lower = 1, upper = 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(file = "~/Machine Learning/Hw1\_Computational\_Problem\_Set/Hw1\_Computational\_Problem\_Set/unc\_salary\_data.csv")

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

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 92 150 ...  
## $ position: Factor w/ 4120 levels "(NC DETECT) Program Director",..: 3597 634 3474 41 3836 2282 41 3369 3361 1136 ...  
## $ 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 20120702 ...  
## $ 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

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

relevant = salaries %>% dplyr::select(name, dept, age, totalsal)  
head(relevant)

## name dept age totalsal  
## 1 AARON, NANCY G Romance Languages 55 46350  
## 2 ABARBANELL, JEFFERY S Kenan-Flagler Business School 57 173000  
## 3 ABARE, BETSY Institute of Marine Sciences 54 38170  
## 4 ABATE, AARON B Medicine Administration 29 50070  
## 5 ABATEMARCO, JODI M School of Education 35 41696  
## 6 ABBOTT-LUNSFORD, SHELBY L Medicine Administration 41 61000

**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 = relevant[which(relevant$totalsal > 200000),]  
head(top\_200)

## name dept age totalsal  
## 41 ADAMS, SASHA D Surgery 42 271000  
## 50 ADAMSON, WILLIAM T Surgery 50 410000  
## 56 ADIMORA, ADAORA A Medicine 58 230614  
## 71 AHALT, STANLEY C Renaissance Computing Inst 60 257940  
## 91 AKINTEMI, OLA B Pediatrics 60 205919  
## 94 AKULIAN, JASON A Medicine 38 230000

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

avg.dept = relevant %>% dplyr::filter(dept %in% c('Statistics and Operations Res',  
 'Philosophy',  
 'Mathematics')) %>% group\_by(dept) %>%  
 summarise(avg.sal = mean(totalsal))  
avg.dept

## # A tibble: 3 x 2  
## dept avg.sal  
## <fct> <dbl>  
## 1 Mathematics 93061.  
## 2 Philosophy 95156.  
## 3 Statistics and Operations Res 100471.

**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).

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

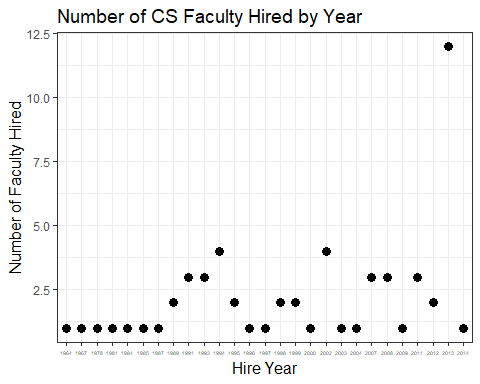
folds <- createFolds(iris$Species, k = 5)  
iris1 = iris[folds$Fold1,]  
iris2 = iris[folds$Fold2,]  
iris3 = iris[folds$Fold3,]  
iris4 = iris[folds$Fold4,]  
iris5 = iris[folds$Fold5,]  
  
  
iris.5fold <- list(iris1, iris2, iris3, iris4, iris5)  
str(iris.5fold)

## List of 5  
## $ :'data.frame': 30 obs. of 5 variables:  
## ..$ Sepal.Length: num [1:30] 5.1 4.9 5 4.8 5.8 5.7 4.4 4.8 4.6 5.3 ...  
## ..$ Sepal.Width : num [1:30] 3.5 3 3.6 3 4 4.4 3.2 3 3.2 3.7 ...  
## ..$ Petal.Length: num [1:30] 1.4 1.4 1.4 1.4 1.2 1.5 1.3 1.4 1.4 1.5 ...  
## ..$ Petal.Width : num [1:30] 0.2 0.2 0.2 0.1 0.2 0.4 0.2 0.3 0.2 0.2 ...  
## ..$ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ :'data.frame': 30 obs. of 5 variables:  
## ..$ Sepal.Length: num [1:30] 4.7 5.4 4.6 4.8 5.2 4.7 5 4.9 4.4 5 ...  
## ..$ Sepal.Width : num [1:30] 3.2 3.7 3.6 3.4 3.5 3.2 3.2 3.6 3 3.3 ...  
## ..$ Petal.Length: num [1:30] 1.3 1.5 1 1.9 1.5 1.6 1.2 1.4 1.3 1.4 ...  
## ..$ Petal.Width : num [1:30] 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.1 0.2 0.2 ...  
## ..$ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ :'data.frame': 30 obs. of 5 variables:  
## ..$ Sepal.Length: num [1:30] 4.6 4.4 4.8 5.4 5.1 5.2 4.8 5.5 5 5 ...  
## ..$ Sepal.Width : num [1:30] 3.4 2.9 3.4 3.9 3.7 3.4 3.1 4.2 3.5 3.5 ...  
## ..$ Petal.Length: num [1:30] 1.4 1.4 1.6 1.3 1.5 1.4 1.6 1.4 1.3 1.6 ...  
## ..$ Petal.Width : num [1:30] 0.3 0.2 0.2 0.4 0.4 0.2 0.2 0.2 0.3 0.6 ...  
## ..$ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ :'data.frame': 30 obs. of 5 variables:  
## ..$ Sepal.Length: num [1:30] 4.6 4.3 5.1 5.7 5.1 5.1 5.4 4.9 5.1 4.5 ...  
## ..$ Sepal.Width : num [1:30] 3.1 3 3.5 3.8 3.8 3.3 3.4 3.1 3.4 2.3 ...  
## ..$ Petal.Length: num [1:30] 1.5 1.1 1.4 1.7 1.5 1.7 1.5 1.5 1.5 1.3 ...  
## ..$ Petal.Width : num [1:30] 0.2 0.1 0.3 0.3 0.3 0.5 0.4 0.2 0.2 0.3 ...  
## ..$ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ :'data.frame': 30 obs. of 5 variables:  
## ..$ Sepal.Length: num [1:30] 5.4 5 4.9 5.4 5 5 5.2 5.5 5.1 5.1 ...  
## ..$ Sepal.Width : num [1:30] 3.9 3.4 3.1 3.4 3 3.4 4.1 3.5 3.8 3.8 ...  
## ..$ Petal.Length: num [1:30] 1.7 1.5 1.5 1.7 1.6 1.6 1.5 1.3 1.9 1.6 ...  
## ..$ Petal.Width : num [1:30] 0.4 0.2 0.1 0.2 0.2 0.4 0.1 0.2 0.4 0.2 ...  
## ..$ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...

**Exercise 12** *(10 pt)*

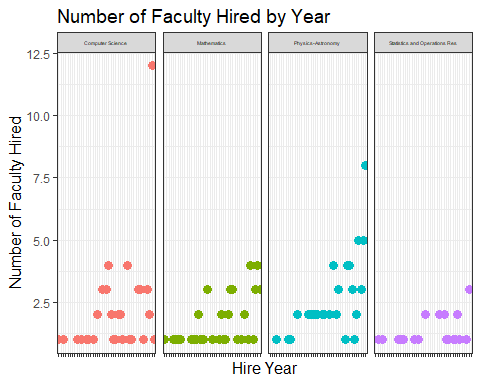
**a** Recall the UNC salary data set. From the salaries data frame plot the number of CS faculty hired per year vs year.

cs = salaries %>% dplyr::filter(dept == 'Computer Science')  
cs$Hire.Year = substr(cs$hiredate, 1, 4)  
cs.summary = cs %>% group\_by(Hire.Year) %>% summarise(Number = n())  
  
ggplot(cs.summary, aes(x= Hire.Year, y = Number)) + geom\_point(size = 3) + xlab('Hire Year') + theme\_bw() +   
 theme(axis.text.x = element\_text(size = 4),  
 title = element\_text(size = 12),  
 axis.title = element\_text(size = 12)) +   
 ylab('Number of Faculty Hired') +   
 ggtitle('Number of CS Faculty Hired by Year')



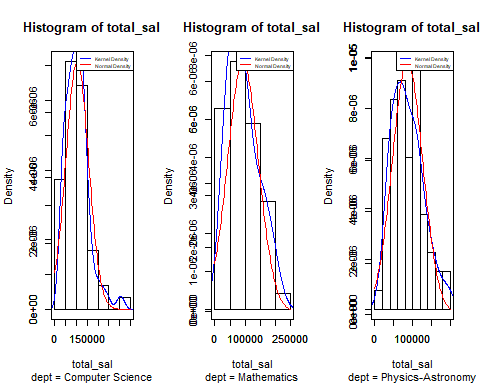
**b** Now add STOR, Math and Physics to the above plot

cs = salaries %>% dplyr::filter(dept %in% c('Computer Science',  
 'Statistics and Operations Res',  
 'Mathematics',  
 'Physics-Astronomy'))  
cs$Hire.Year = substr(cs$hiredate, 1, 4)  
cs.summary = cs %>% group\_by(dept, Hire.Year) %>% summarise(Number = n())  
  
#cs.summary$Hire.Year = substr(cs.summary$Hire.Year, 3, 4)  
  
ggplot(cs.summary, aes(x= Hire.Year, y = Number,  
 colour = dept)) + geom\_point(size = 3) + xlab('Hire Year') + theme\_bw() +   
 theme(axis.text.x = element\_blank(),  
 title = element\_text(size = 12),  
 legend.position = 'none',  
 strip.text.x = element\_text(size = 4),  
 axis.title = element\_text(size = 12)) + facet\_grid(. ~ `dept`) +   
 ylab('Number of Faculty Hired') + ggtitle('Number of Faculty Hired by Year')



**Exercise 13.** *(15 pt)* The following code generates the ensuing plot about Sepal.Length in iris. Either modify the code above or use your own code to obtain similar plots with histograms, kernel density plots and normal density plots for the salary of faculty in CS, Math and Physics from UNC salary data.

non.stor = salaries %>% dplyr::filter(dept %in% c('Computer Science',  
 'Mathematics',  
 'Physics-Astronomy'))  
opar <- par(mfrow = c(1,3))  
  
non.stor$dept = factor(non.stor$dept)  
for(l in levels(non.stor$dept))  
{  
 total\_sal <- subset(non.stor, dept == l, select = totalsal)[[1]]  
 h <- hist(total\_sal, sub = paste("dept =", l), freq = FALSE)  
  
 par(new = TRUE) # add to the current plot  
 # Empirical density curve  
 lines(density(total\_sal),  
 xlim = range(h$breaks), # to match the plotting range  
 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(total\_sal), sd = sd(total\_sal)),  
 xlim = range(h$breaks), # to match the plotting range  
 col = "red",  
 main = "", sub = "", xlab = "", ylab = "" # to supress labels  
 )  
  
 legend("topright",  
 legend = c("Kernel Density", "Normal Density"),  
 col = c("blue", "red"), lty = 1, cex = 0.5)  
}



par(opar)