Homework 1

Mamata Anil Parab

```
library(MVA)
library(norm)
library(scatterplot3d)
library(KernSmooth)
library(ResourceSelection)
library(MASS)
```

Problem 1: Answer the following questions.

a. How many rows are in an n × 3 matrix? How many columns? Choose a number for n and give an example in R.

Here we consider **n=3**

```
mat= matrix(seq(1,9), byrow=T, nrow=3)
mat
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
```

Number of rows:

```
nrow(mat)
```

```
## [1] 3
```

Number of columns:

```
ncol(mat)
```

```
## [1] 3
```

OR we can also use **dim** function to see dimensions of matrix. The first number of dimension indicates number of rows whereas second number indicates number of columns

```
dim(mat)
```

```
## [1] 3 3
```

So, in **3x3** matrix, 3 columns and 3 rows are present.

b. Give an example of a 3 $\times$ 2 matrix in R. The **3x2** matrix is:

```
mat2= matrix(c(1, 2, 8, 9, 5, 6), byrow=T, nrow=3)
mat2
```

```
##      [,1] [,2]
## [1,]    1    2
## [2,]    8    9
## [3,]    5    6
```

```
dim(mat2)
```

```
## [1] 3 2
```

Problem 2: You can use R to help you answer the following questions, but you must articulate your explanations in written form. For each of the following three expressions answer the following: (1) Is the expression solvable? (2) If yes, provide a solution; if not, explain why it is not solvable.

a.

$$\begin{bmatrix} 2 & -1 \\ 1 & 2 \end{bmatrix} * \begin{bmatrix} 1 & 1 & 2 \\ 2 & 0 & 1 \end{bmatrix}$$

In general, matrix multiplication **AxB** of two matrices **A** and **B** is possible if the number of columns of **A** is equal to the number of rows of **B**.

```
A= matrix(c(2,-1,1,2), byrow=T, nrow=2)
A
```

```
##      [,1] [,2]
## [1,]    2   -1
## [2,]    1    2
```

```
B= matrix(c(1,1,2,2,0,1), byrow=T, nrow=2)
B
```

```
##      [,1] [,2] [,3]
## [1,]    1    1    2
## [2,]    2    0    1
```

The dimension of A and B are:

```
dim(A)
```

```
## [1] 2 2
```

```
dim(B)
```

```
## [1] 2 3
```

```
ncol(A)== nrow(B)
```

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

Here number of columns of A is equal to number of rows of B. So the given expression is **solvable**.

```
A %*% B
```

```
##      [,1] [,2] [,3]
## [1,]    0    2    3
## [2,]    5    1    4
```

b.

$$\begin{bmatrix} 1 & 0 \\ 2 & 1 \\ -2 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 1 & 2 \\ 2 & 0 & 1 \end{bmatrix}$$

In general, matrix multiplication **AxB** of two matrices **A** and **B** is possible if the number of columns of **A** is equal to the number of rows of **B**.

```
A= matrix(c(1,0,2,1,-2,1), byrow=T, nrow=3)
A
```

```
##      [,1] [,2]
## [1,]    1    0
## [2,]    2    1
## [3,]   -2    1
```

```
B= matrix(c(1,1,2,2,0,1), byrow=T, nrow=2)
B
```

```
##      [,1] [,2] [,3]
## [1,]    1    1    2
## [2,]    2    0    1
```

The dimension of A and B are:

```
dim(A)
```

```
## [1] 3 2
```

```
dim(B)
```

```
## [1] 2 3
```

```
ncol(A)== nrow(B)
```

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

Here number of columns of A is equal to number of rows of B.So the given expression is **solvable**.

```
A %*% B
```

```
##      [,1] [,2] [,3]
## [1,]    1    1    2
## [2,]    4    2    5
## [3,]    0   -2   -3
```

c.

$$\begin{bmatrix} 1 & 0 \\ 2 & 1 \\ -2 & 1 \end{bmatrix} * \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$$

In general, matrix multiplication **AxB** of two matrices **A** and **B** is possible if the number of columns of **A** is equal to the number of rows of **B**.

```
A= matrix(c(1,0,2,1,-2,1), byrow=T, nrow=3)
A
```

```
##      [,1] [,2]
## [1,]    1    0
## [2,]    2    1
## [3,]   -2    1
```

```
B= matrix(c(1,-1,0), ncol=1)
B
```

```
##      [,1]
## [1,]    1
## [2,]   -1
## [3,]    0
```

The dimension of A and B are:

```
dim(A)
```

```
## [1] 3 2
```

```
dim(B)
```

```
## [1] 3 1
```

```
ncol(A)== nrow(B)
```

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

Here number of columns of A is not equal to number of rows of B. So the given expression is **not solvable**.

Problem 3: You can use R to help you answer the following questions, but you must articulate your explanations in written form. For each of the following three expressions answer the following: (1) Is the expression solvable? (2) If yes, provide a solution; if not, explain why it is not solvable.

a. $A * A^{-1}$ where $A=$

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

To solve this question, we first calculate **determinant** of matrix. If the determinant of matrix is zero then matrix will not have inverse. Such a matrix is called as **Singular** matrix.

```
A= matrix(c(2,1,1,2), byrow=T, nrow=2)
A
```

```
##      [,1] [,2]
## [1,]    2    1
## [2,]    1    2
```

```
det(A)
```

```
## [1] 3
```

As determinant of A=3 $(\mathbf{det(A)} \neq \mathbf{0})$, the given expression is **solvable**.

We can find an inverse as:

```
solve(A)
```

```
##            [,1]       [,2]
## [1,]  0.6666667 -0.3333333
## [2,] -0.3333333  0.6666667
```

A **x** A^{-1} can be calculated as:

```
A %*% solve(A)
```

```
##      [,1] [,2]
## [1,]    1    0
## [2,]    0    1
```

b. A * A^{-1} where A=

$$\begin{bmatrix} 2 & 0 \\ 1 & 0 \end{bmatrix}$$

To solve this question, we first calculate **determinant** of matrix. If the determinant of matrix is zero then it will not have inverse. Such a matrix is called as **Singular** matrix.

```
A= matrix(c(2,0,1,0), byrow=T, nrow=2)
A
```

```
##      [,1] [,2]
## [1,]    2    0
## [2,]    1    0
```

```
det(A)
```

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

As determinant of A=0 $(\mathbf{det(A) = 0})$, the given expression is **not solvable**.

Problem 4: Download women's track record dataset as well as the men's track record dataset. These datasets hold data on men's and women's records for various races in various countries. Note that the columns of the two datasets do not match exactly.

```
women <- read.csv("https://raw.githubusercontent.com/EricBrownTTU/ISQS6350/main/womens_track.csv")
head(women)
```

```
##    m100  m200  m400 m800 m1500 m3000 marathon  country
## 1 11.61 22.94 54.50 2.15  4.43  9.79   178.52 argentin
## 2 11.20 22.35 51.08 1.98  4.13  9.08   152.37 australi
## 3 11.43 23.09 50.62 1.99  4.22  9.34   159.37  austria
## 4 11.41 23.04 52.00 2.00  4.14  8.88   157.85  belgium
## 5 11.46 23.05 53.30 2.16  4.58  9.81   169.98  bermuda
## 6 11.31 23.17 52.80 2.10  4.49  9.77   168.75   brazil
```

```
men <- read.csv("https://raw.githubusercontent.com/EricBrownTTU/ISQS6350/main/mens_track.csv")
head(men)
```

```
##    m100  m200  m400 m800 m1500 m3000 mystery marathon  country
## 1 10.39 20.81 46.84 1.81  3.70 14.04   29.36   137.72 argentin
## 2 10.31 20.06 44.84 1.74  3.57 13.28   27.66   128.30 australi
## 3 10.44 20.81 46.82 1.79  3.60 13.26   27.72   135.90  austria
## 4 10.34 20.68 45.04 1.73  3.60 13.22   27.45   129.95  belgium
## 5 10.28 20.58 45.91 1.80  3.75 14.68   30.55   146.62  bermuda
## 6 10.22 20.43 45.21 1.73  3.66 13.62   28.62   133.13   brazil
```

a. Report the covariance and correlation matrix among the women's records. Which variables are the most correlated? Why?

Covariance is similar to correlation, except the data are not normalized when the covariance is computed. As a result, the covariance is represented in data-dependent units rather than being converted to a standardized scale of -1 to 1.

```
round(cov(women[seq(1,7)]),6)
```

```
##              m100      m200      m400     m800    m1500     m3000   marathon
## m100     0.204494  0.478714  1.010955 0.035613 0.109493  0.276485   9.444360
## m200     0.478714  1.234455  2.550142 0.087063 0.257937  0.650161  23.178626
## m400     1.010955  2.550142  7.173488 0.260412 0.701453  1.716905  57.492462
## m800     0.035613  0.087063  0.260412 0.011712 0.032437  0.077041   2.566373
## m1500    0.109493  0.257937  0.701453 0.032437 0.110507  0.265582   8.880786
## m3000    0.276485  0.650161  1.716905 0.077041 0.265582  0.679529  22.571667
## marathon 9.444360 23.178626 57.492462 2.566373 8.880786 22.571667 925.957204
```

```
round(cor(women[seq(1,7)]),6)
```

```
##              m100     m200     m400     m800    m1500    m3000 marathon
## m100     1.000000 0.952791 0.834692 0.727689 0.728371 0.741699 0.686336
## m200     0.952791 1.000000 0.856962 0.724060 0.698364 0.709871 0.685575
## m400     0.834692 0.856962 1.000000 0.898405 0.787842 0.777637 0.705424
## m800     0.727689 0.724060 0.898405 1.000000 0.901614 0.863565 0.779292
## m1500    0.728371 0.698364 0.787842 0.901614 1.000000 0.969169 0.877933
## m3000    0.741699 0.709871 0.777637 0.863565 0.969169 1.000000 0.899837
## marathon 0.686336 0.685575 0.705424 0.779292 0.877933 0.899837 1.000000
```

Among all the variables, the correlation coefficient between **m1500** and **m3000** is high i.e. **0.9691690**, indicating a positive, high correlation (dependency) between two records (m1500 and m3000). Similarly, the covariance between **marathon** and **m400** is high i.e. **57.492462**, indicating a positive and high dependency between two records (marathon and m400)

b. Report the covariance and correlation matrix among the men's records. Which variables are the most correlated? Why?

Covariance is similar to correlation, except the data are not normalized when the covariance is computed. As a result, the covariance is represented in data-dependent units rather than being converted to a standardized scale of -1 to 1.

```
round(cov(men[seq(1,8)]),5)
```

```
##             m100    m200    m400    m800   m1500   m3000  mystery marathon
## m100     0.12350 0.20902 0.43070 0.01692 0.03837 0.17441  0.40185  1.68601
## m200     0.20902 0.41557 0.79906 0.03312 0.07789 0.35914  0.81171  3.54621
## m400     0.43070 0.79906 2.12290 0.08074 0.18974 0.90888  2.07342  9.47786
## m800     0.01692 0.03312 0.08074 0.00406 0.00912 0.04406  0.10005  0.47390
## m1500    0.03837 0.07789 0.18974 0.00912 0.02431 0.11593  0.26344  1.24516
## m3000    0.17441 0.35914 0.90888 0.04406 0.11593 0.64186  1.41155  6.89105
## mystery  0.40185 0.81171 2.07342 0.10005 0.26344 1.41155  3.26789 15.73218
## marathon 1.68601 3.54621 9.47786 0.47390 1.24516 6.89105 15.73218 85.13815
```

```
round(cor(men[seq(1,8)]),5)
```

```
##             m100    m200    m400    m800   m1500   m3000 mystery marathon
## m100     1.00000 0.92264 0.84115 0.75603 0.70024 0.61946 0.63254  0.51995
## m200     0.92264 1.00000 0.85073 0.80663 0.77495 0.69538 0.69654  0.59618
## m400     0.84115 0.85073 1.00000 0.87017 0.83527 0.77861 0.78720  0.70499
## m800     0.75603 0.80663 0.87017 1.00000 0.91804 0.86359 0.86905  0.80648
## m1500    0.70024 0.77495 0.83527 0.91804 1.00000 0.92811 0.93470  0.86555
## m3000    0.61946 0.69538 0.77861 0.86359 0.92811 1.00000 0.97464  0.93219
## mystery  0.63254 0.69654 0.78720 0.86905 0.93470 0.97464 1.00000  0.94318
## marathon 0.51995 0.59618 0.70499 0.80648 0.86555 0.93219 0.94318  1.00000
```

Among all the variables, the correlation coefficient between **m3000** and **mystery** is high i.e. **0.97464**,indicating a positive, high correlation (dependency) between two records. Similarly the covariance between **mystery** and **marathon** is high i.e. **15.73218**, indicating a positive and high dependency between two records

c. Find the Euclidean distance matrix for the first five rows of the women's data combined with the first five rows of the men's data for columns which appear in both datasets. Present your matrix to 2 decimal places.

First five rows of women data are:

```
w_data= women[seq(1,5),seq(1,7)]
w_data
```

```
##    m100  m200  m400 m800 m1500 m3000 marathon
## 1 11.61 22.94 54.50 2.15  4.43  9.79   178.52
## 2 11.20 22.35 51.08 1.98  4.13  9.08   152.37
## 3 11.43 23.09 50.62 1.99  4.22  9.34   159.37
## 4 11.41 23.04 52.00 2.00  4.14  8.88   157.85
## 5 11.46 23.05 53.30 2.16  4.58  9.81   169.98
```

```
m_data= men[seq(1,5), c(seq(1,6),8)]
m_data
```

```
##    m100  m200  m400 m800 m1500 m3000 marathon
## 1 10.39 20.81 46.84 1.81  3.70 14.04   137.72
## 2 10.31 20.06 44.84 1.74  3.57 13.28   128.30
## 3 10.44 20.81 46.82 1.79  3.60 13.26   135.90
## 4 10.34 20.68 45.04 1.73  3.60 13.22   129.95
## 5 10.28 20.58 45.91 1.80  3.75 14.68   146.62
```

```
combined_data= rbind(w_data, m_data)
combined_data
```

```
##     m100  m200  m400 m800 m1500 m3000 marathon
## 1  11.61 22.94 54.50 2.15  4.43  9.79   178.52
## 2  11.20 22.35 51.08 1.98  4.13  9.08   152.37
## 3  11.43 23.09 50.62 1.99  4.22  9.34   159.37
## 4  11.41 23.04 52.00 2.00  4.14  8.88   157.85
## 5  11.46 23.05 53.30 2.16  4.58  9.81   169.98
## 6  10.39 20.81 46.84 1.81  3.70 14.04   137.72
## 7  10.31 20.06 44.84 1.74  3.57 13.28   128.30
## 8  10.44 20.81 46.82 1.79  3.60 13.26   135.90
## 9  10.34 20.68 45.04 1.73  3.60 13.22   129.95
## 10 10.28 20.58 45.91 1.80  3.75 14.68   146.62
```

It is useful to standardize the data before calculating the distance, to avoid scale discrepancies.

```
scale_cdata= scale(combined_data)
round(scale_cdata,5)
```

```
##           m100     m200     m400     m800    m1500    m3000 marathon
##  [1,]  1.25968  0.95921  1.50194  1.44800  1.22612 -0.74960  1.69797
##  [2,]  0.54534  0.48721  0.55159  0.40051  0.42299 -1.05407  0.15955
##  [3,]  0.94607  1.07922  0.42377  0.46213  0.66393 -0.94257  0.57136
##  [4,]  0.91122  1.03922  0.80724  0.52375  0.44976 -1.13983  0.48194
##  [5,]  0.99834  1.04722  1.16849  1.50962  1.62769 -0.74102  1.19555
##  [6,] -0.86592 -0.74481 -0.62662 -0.64698 -0.72818  1.07294 -0.70232
##  [7,] -1.00531 -1.34482 -1.18238 -1.07830 -1.07620  0.74702 -1.25650
##  [8,] -0.77881 -0.74481 -0.63218 -0.77021 -0.99589  0.73845 -0.80939
##  [9,] -0.95304 -0.84881 -1.12681 -1.13992 -0.99589  0.72129 -1.15943
## [10,] -1.05758 -0.92881 -0.88505 -0.70860 -0.59432  1.34739 -0.17873
## attr(,"scaled:center")
##     m100     m200     m400     m800    m1500    m3000 marathon
##   10.887   21.741   49.095    1.915    3.972   11.538  149.658
## attr(,"scaled:scale")
##       m100        m200        m400        m800       m1500       m3000    marathon
##  0.5739541   1.2499818   3.5986703   0.1622926   0.3735357   2.3319177 16.9979828
```

Euclidean distance

```
dist_s = dist(scale_cdata, method = "euclidean")
round((dist_s),2)
```

```
##       1    2    3    4    5    6    7    8    9
## 2  2.42
## 3  1.97 0.88
## 4  1.92 0.79 0.50
## 5  0.78 2.18 1.74 1.78
## 6  5.41 3.55 4.07 4.18 5.20
## 7  6.35 4.30 4.88 4.98 6.12 1.19
## 8  5.47 3.49 4.05 4.13 5.28 0.47 1.01
## 9  6.09 4.02 4.56 4.67 5.85 0.96 0.52 0.74
## 10 5.52 3.85 4.30 4.45 5.33 0.71 1.47 1.05 1.33
```

d. Given your solution to part (c), which two entries are the most similar? Which are the most dissimilar? Explain how you determined this.

In the euclidean distance matrix, similarity and dissimilarity depends on the distance between two points or entities.

If the distance between two entities is less then they are more likely to be similar or more likely possess similar characteristics. But if the distance between two entities is higher, then they are more likely to be dissimilar.

As shown in (c) the euclidean distance between the entities 8 and 6 is **0.47** which is the smallest and indicating two entities are most similar in characteristics. Similarly, the euclidean distance between entities 1 and 7 is **6.35** which is largest and indicating two points are most dissimilar in characteristics.

Problem 5: Answer the following questions.

a. Convert the following covariance matrix into the corresponding correlation matrix using R

$$Cov = \begin{bmatrix} 3.8778 & 2.8110 & 3.1480 & 3.5062 \\ 2.8110 & 2.1210 & 2.2669 & 2.5690 \\ 3.1480 & 2.2669 & 2.6550 & 2.8341 \\ 3.5062 & 2.5690 & 2.8341 & 3.2352 \end{bmatrix}$$

The covariance matrix cov is:

```r
Cov= matrix(c(3.8778, 2.8110, 3.1480, 3.5062,
              2.8110, 2.1210, 2.2669, 2.5690,
              3.1480, 2.2669, 2.6550, 2.8341,
              3.5062, 2.5690, 2.8341, 3.2352), byrow=T, nrow=4)

Cov
```

```
##        [,1]   [,2]   [,3]   [,4]
## [1,] 3.8778 2.8110 3.1480 3.5062
## [2,] 2.8110 2.1210 2.2669 2.5690
## [3,] 3.1480 2.2669 2.6550 2.8341
## [4,] 3.5062 2.5690 2.8341 3.2352
```

The corresponding correlation matrix is:

```r
cov2cor(Cov)
```

```
##           [,1]      [,2]      [,3]      [,4]
## [1,] 1.0000000 0.9801619 0.9810921 0.9899048
## [2,] 0.9801619 1.0000000 0.9552780 0.9807159
## [3,] 0.9810921 0.9552780 1.0000000 0.9670131
## [4,] 0.9899048 0.9807159 0.9670131 1.0000000
```

b. Show how to find the correlation between the first two variables based on the covariance matrix using simple algebra.

The correlation of first two variables is using simple algebra:

```
Cov[1,2]/(sqrt(Cov[1,1])*sqrt(Cov[2,2]))
```

```
## [1] 0.9801619
```

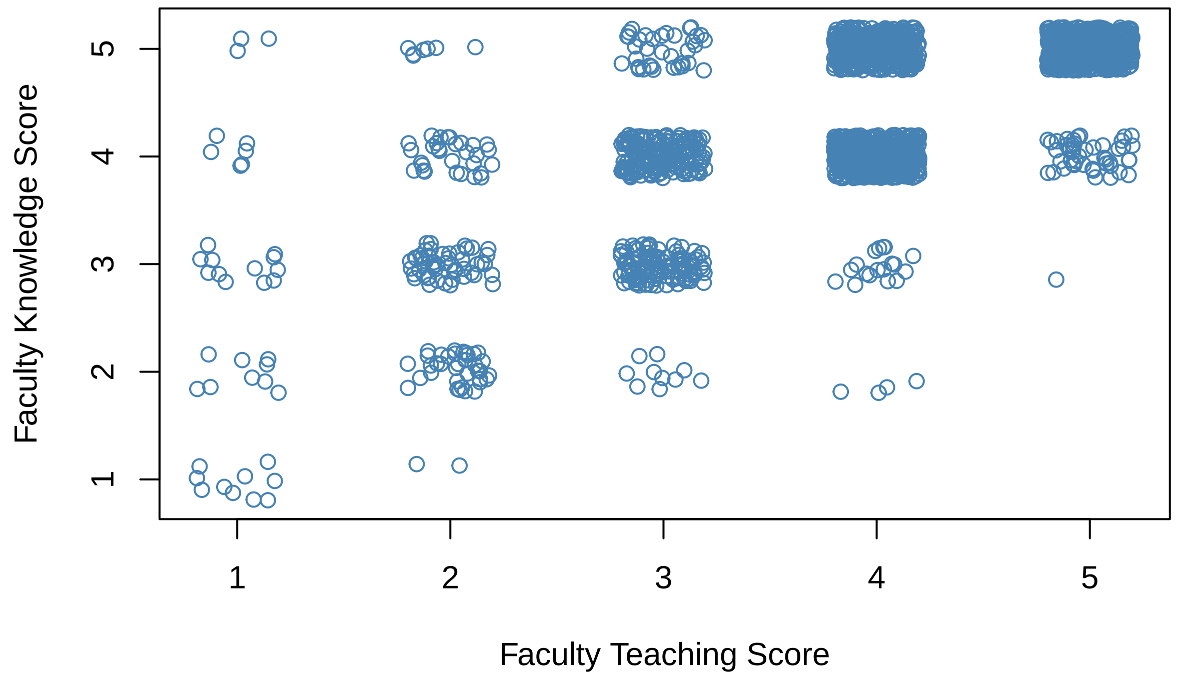
Problem 6: Use the TTU graduate student exit survey data

```
grad <- read.csv("https://raw.githubusercontent.com/EricBrownTTU/ISQS6350/main/pgs.csv")
```

a. Make a scatterplot of two variables: the score of faculty teaching "FacTeaching" and faculty knowledge "FacKnowledge." Consider using the jitter() function in this analysis, and explain why it may be useful.

jittering introduces random noise to data, which may be useful for plotting data in a scatterplot. We can gain a clearer understanding of the real underlying relationship between two variables in a dataset by utilizing the jitter function.

```
plot(jitter(grad$FacTeaching), jitter(grad$FacKnowledge), col = 'steelblue',
     ylab = "Faculty Knowledge Score",
     xlab = "Faculty Teaching Score")
```



b. Create a new data frame for three variables: "FacTeaching", "FacKnowledge", and "Housing".

We will create subset of **grad** data as:

```
subcol =c("FacTeaching", "FacKnowledge", "Housing")
subgrad= grad[subcol]
head(subgrad)
```

```
##   FacTeaching FacKnowledge Housing
## 1           3            3       4
## 2           3            4       3
## 3           4            4       4
## 4           3            3       2
## 5           4            4      NA
## 6           4            5       4
```

c. Find a correlation matrix of the data in part (b). If there are missing values in your data, report how many missing values there are and where they appear. Then estimate the correlation matrix via:

i. Complete-case analysis

ii. Available-case analysis

iii. Maximum likelihood estimation

Is there a noticeable difference among the three methods? Which method do you suggest?

```
cor(subgrad)
```

```
##              FacTeaching FacKnowledge Housing
## FacTeaching            1           NA      NA
## FacKnowledge          NA            1      NA
## Housing               NA           NA       1
```

```
sum(is.na(subgrad))
```

```
## [1] 350
```

There are **350** missing values in **subgrad**.

The rows where missing values are present can be determine by:

```
subgrad_na = subgrad[!complete.cases(subgrad),] # OR subgrad[is.na(subgrad),]
head(subgrad_na, 20) #Here we are showing only first 20 rows
```

```
##    FacTeaching FacKnowledge Housing
## 5            4            4      NA
## 8            5            5      NA
## 9            4            3      NA
## 13           4            5      NA
## 14           4            5      NA
## 15           3            4      NA
## 16           4            5      NA
## 18           4            5      NA
## 27           4            5      NA
## 32           5            5      NA
## 38           5            5      NA
## 42           3            4      NA
## 45           5            5      NA
## 48           4            4      NA
## 55           4            5      NA
## 62           2            2      NA
## 66          NA           NA      NA
## 74           4            4      NA
## 75           4            4      NA
## 76           4            4      NA
```

1) Complete Case Analysis

```
cor(subgrad, use = "complete.obs")
```

```
##              FacTeaching FacKnowledge   Housing
## FacTeaching    1.0000000    0.7113826 0.1536942
## FacKnowledge   0.7113826    1.0000000 0.2108483
## Housing        0.1536942    0.2108483 1.0000000
```

2) Available-case analysis

```
cor(subgrad, use = "pairwise.complete.obs")
```

```
##              FacTeaching FacKnowledge   Housing
## FacTeaching    1.0000000    0.7109232 0.1556915
## FacKnowledge   0.7109232    1.0000000 0.2102034
## Housing        0.1556915    0.2102034 1.0000000
```

3) Maximum likelihood estimation

```
library(norm)
s <- prelim.norm(as.matrix(subgrad))
x <- em.norm(s)
```

```
## Iterations of EM:
## 1...2...3...4...5...6...
```

```
getparam.norm(s,x,corr=T)
```

```
## $mu
## [1] 3.931453 4.291835 3.477774
##
## $sdv
## [1] 0.9460864 0.8140599 0.9765425
##
## $r
##           [,1]      [,2]      [,3]
## [1,] 1.0000000 0.7120454 0.1541005
## [2,] 0.7120454 1.0000000 0.2103328
## [3,] 0.1541005 0.2103328 1.0000000
```

```
getparam.norm(s,x,corr=F)
```

```
## $mu
## [1] 3.931453 4.291835 3.477774
##
## $sigma
##           [,1]      [,2]      [,3]
## [1,] 0.8950796 0.5483968 0.1423725
## [2,] 0.5483968 0.6626936 0.1672070
## [3,] 0.1423725 0.1672070 0.9536352
```

The output for **Complete Case Analysis**, **Available-case analysis** and **Maximum likelihood estimation** are almost equal. According to me, **Maximum likelihood estimation** is the most appropriate method. In **Maximum likelihood estimation**, the likelihood is computed separately for complete case on some variables and those with complete data on all variables. These two likelihoods are then maximized together to find the estimates.

Problem 7: Consider the fish dataset. This data describe a set of measurements made on fish caught in a single lake in Finland. Use the bivariate boxplot on the pairs of variables (weight, height3) and (weight,hgtpct) to identify any outliers. Calculate the correlation between each pair of variables using all complete cases of data and the data with any identified outliers removed. Comment on the results.

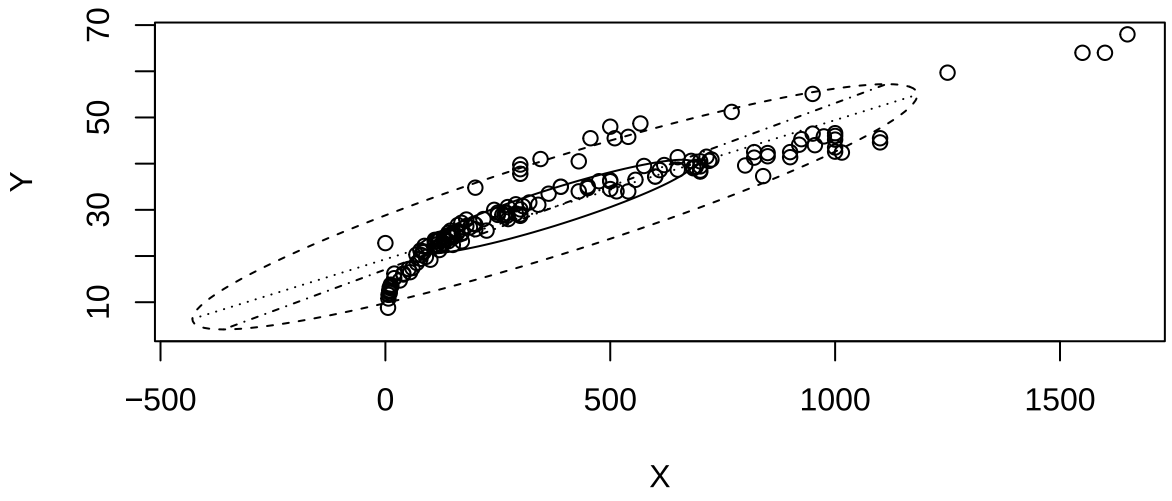
Dowloading the dataset as:

```
fish <- read.csv("https://raw.githubusercontent.com/EricBrownTTU/ISQS6350/main/fish.csv")
head(fish)
```

```
##   obs species weight length1 length2 length3 hgtpct widpct sex
## 1   1       1    242    23.2    25.4    30.0   38.4   13.4  NA
## 2   2       1    290    24.0    26.3    31.2   40.0   13.8  NA
## 3   3       1    340    23.9    26.5    31.1   39.8   15.1  NA
## 4   4       1    363    26.3    29.0    33.5   38.0   13.3  NA
## 5   5       1    430    26.5    29.0    34.0   36.6   15.1  NA
## 6   6       1    450    26.8    29.7    34.7   39.2   14.2  NA
```

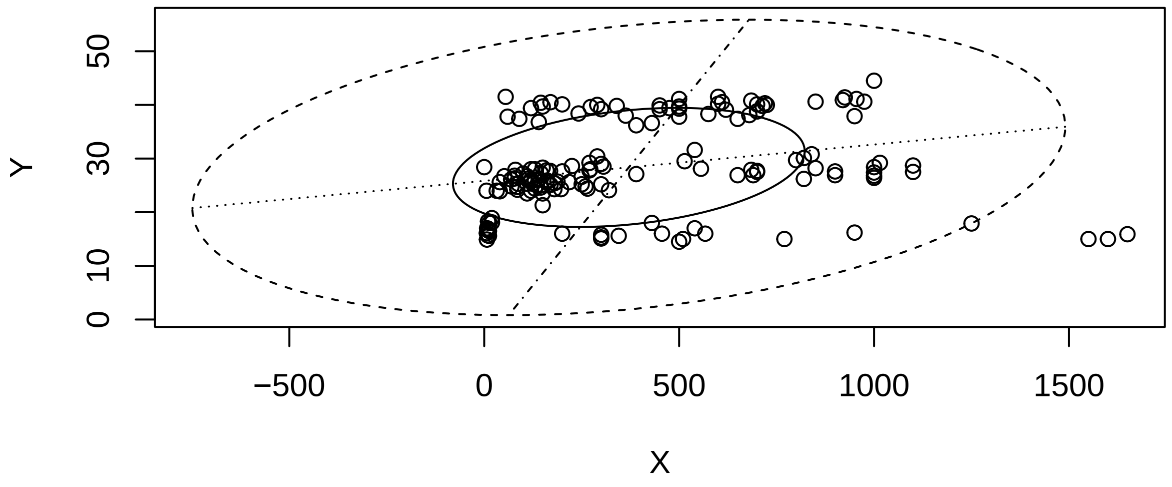
The bivariate boxplot for the pairs of variables (weight, height3) is:

```
fish1 = na.omit(fish[, c('weight', 'length3')])
bvbox(fish1)
```



The bivariate boxplot for the pairs of variables (weight, hgtpct) is:

```
fish2 = na.omit(fish[, c('weight', 'hgtpct')])
bvbox(fish2)
```



The correlation between weight and length3

```r
w1 = fish1[fish$weight<1100, ]
cor(w1[complete.cases(w1),])
```

```
##            weight   length3
## weight  1.0000000 0.9081017
## length3 0.9081017 1.0000000
```

The correlation between weight and hgtpct

```r
w2 = fish2[fish$weight<1500, ]
cor(w2[complete.cases(w2),])
```

```
##           weight    hgtpct
## weight 1.0000000 0.2793218
## hgtpct 0.2793218 1.0000000
```

Problem 8: The Swiss banknote dataset contains measurements on 200 Swiss banknotes: 100 genuine and 100 counterfeit. The variables are the status of the "note", length of bill, width of left edge, width of right edge, bottom margin width, and top margin width. All measurements are in millimeters. Read the data and pick the variables: "note", "top__margin" and "diag__length".

Reading the dataset

```r
swiss <- read.csv("https://raw.githubusercontent.com/EricBrownTTU/ISQS6350/main/swiss.csv")
head(swiss)
```

```
##   note length left_width right_width bottom_margin top_margin diag_length
## 1 real  214.8      131.0       131.1           9.0        9.7       141.0
## 2 real  214.6      129.7       129.7           8.1        9.5       141.7
## 3 real  214.8      129.7       129.7           8.7        9.6       142.2
## 4 real  214.8      129.7       129.6           7.5       10.4       142.0
## 5 real  215.0      129.6       129.7          10.4        7.7       141.8
## 6 real  215.7      130.8       130.5           9.0       10.1       141.4
```

Subset of swiss dataset is:

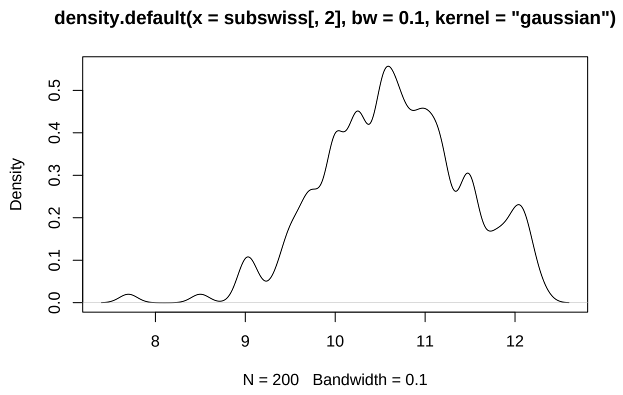
```r
subswiss= swiss[, c("note", "top_margin", "diag_length")]
head(subswiss)
```

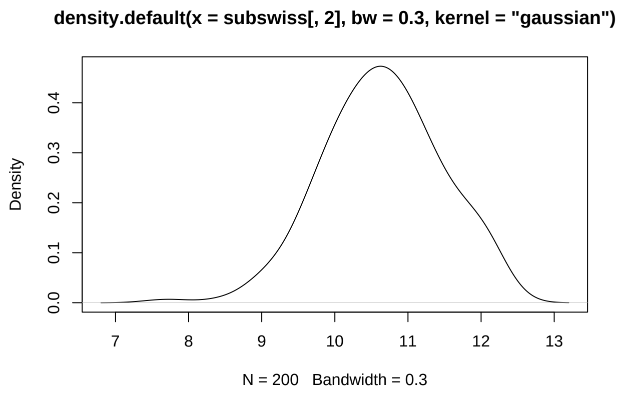
```
##   note top_margin diag_length
## 1 real        9.7       141.0
## 2 real        9.5       141.7
## 3 real        9.6       142.2
## 4 real       10.4       142.0
## 5 real        7.7       141.8
## 6 real       10.1       141.4
```

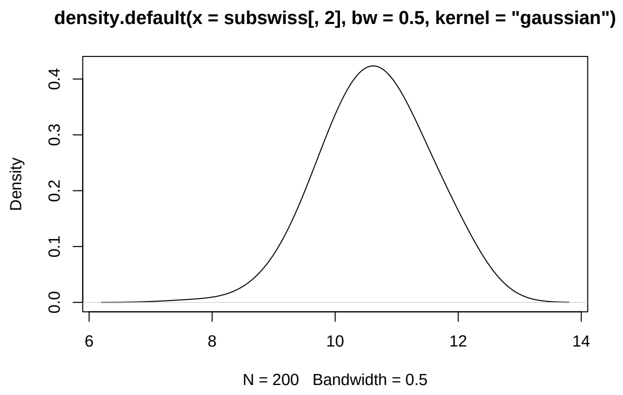
a. Construct separate univariate kernel estimates (using the Gaussian kernel) of the distributions of the two continuous variables. Experiment with bandwidths to get appropriate graphs.

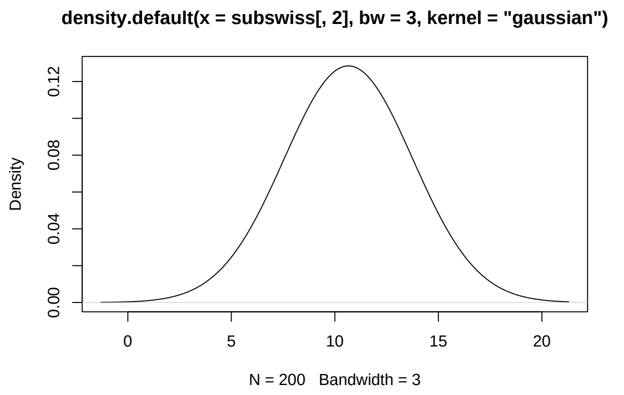
Univariate Gaussian kernel estimation for top_margin for different bandwidths

```r
plot(density(subswiss[,2], kernel = "gaussian", bw = 0.1))
plot(density(subswiss[,2], kernel = "gaussian", bw = 0.3))
plot(density(subswiss[,2], kernel = "gaussian", bw = 0.5))
plot(density(subswiss[,2], kernel = "gaussian", bw = 3))
```



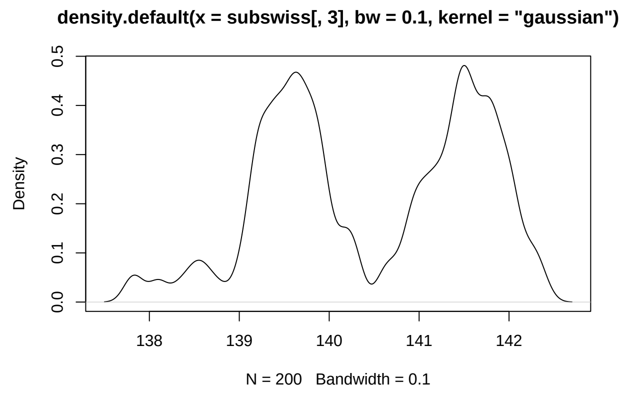


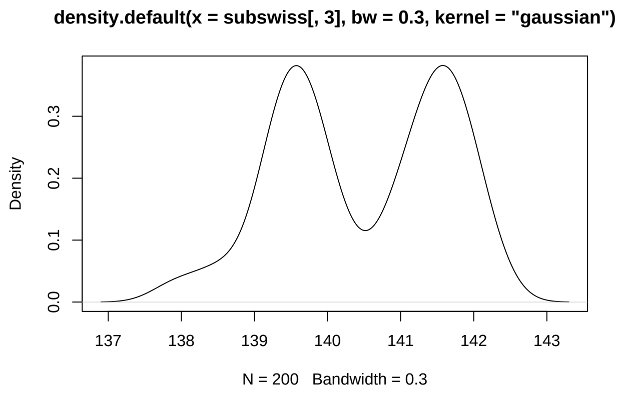


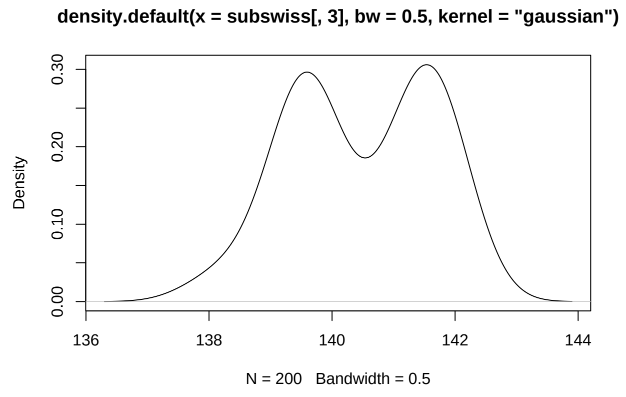


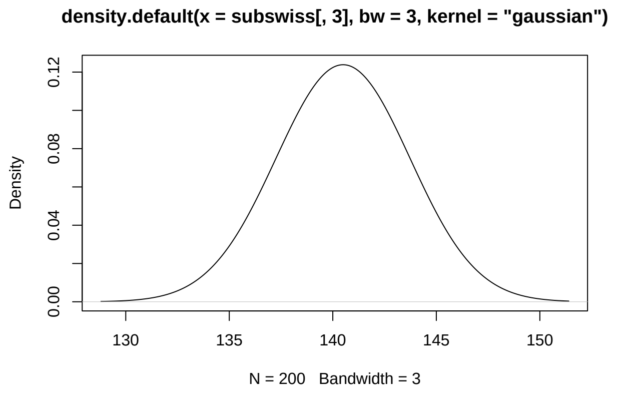
Univariate Gaussian kernel estimation for diag_length for different bandwidths

```r
plot(density(subswiss[,3], kernel = "gaussian", bw = 0.1))
plot(density(subswiss[,3], kernel = "gaussian", bw = 0.3))
plot(density(subswiss[,3], kernel = "gaussian", bw = 0.5))
plot(density(subswiss[,3], kernel = "gaussian", bw = 3))
```







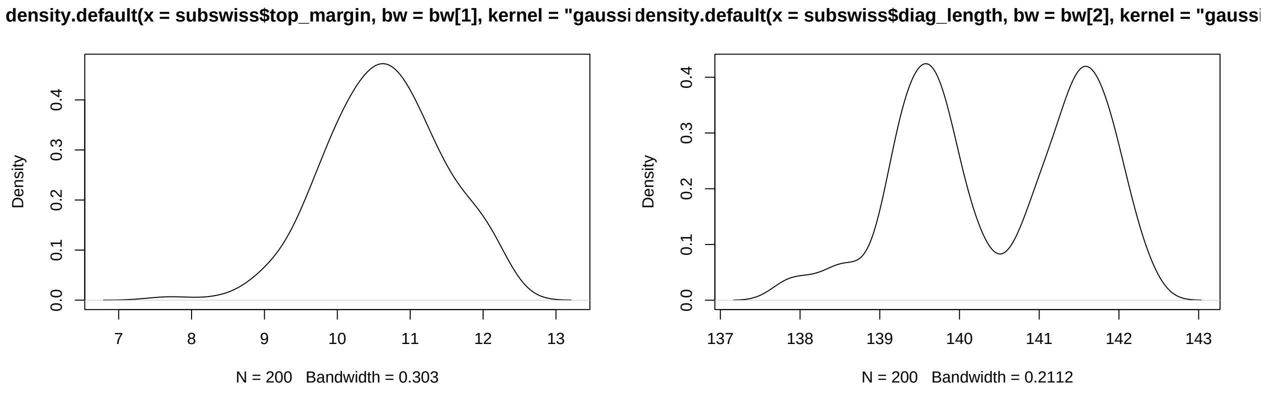


We can find appropriate bandwidths using the dpik function

```r
bw <- c(dpik(subswiss$top_margin), dpik(subswiss$diag_length))
bw
```

```
## [1] 0.3029760 0.2112253
```

```r
plot(density(subswiss$top_margin, kernel = "gaussian", bw = bw[1]))
plot(density(subswiss$diag_length, kernel = "gaussian", bw = bw[2]))
```



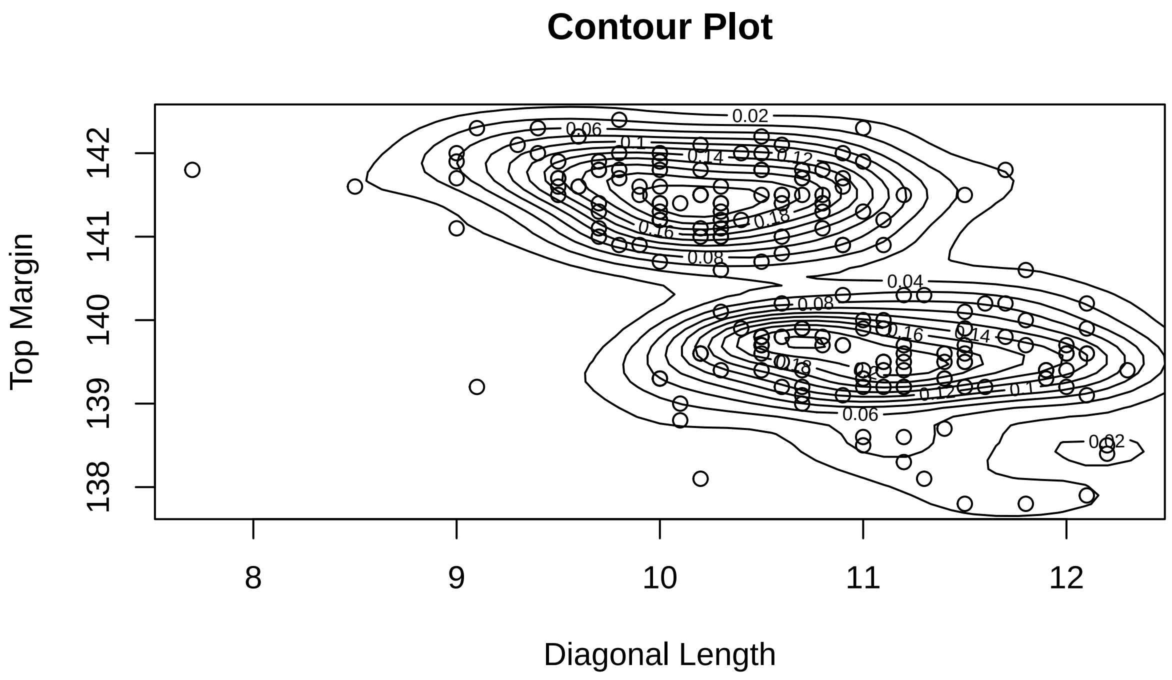
b. Using the bivariate Gaussian kernel, estimate the bivariate density of the two variables using (1) a contour plot and (2) a perspective plot. Use the bandwidths chosen in part (a).

Contour plot

```
a=subswiss[,c(2,3)]
```

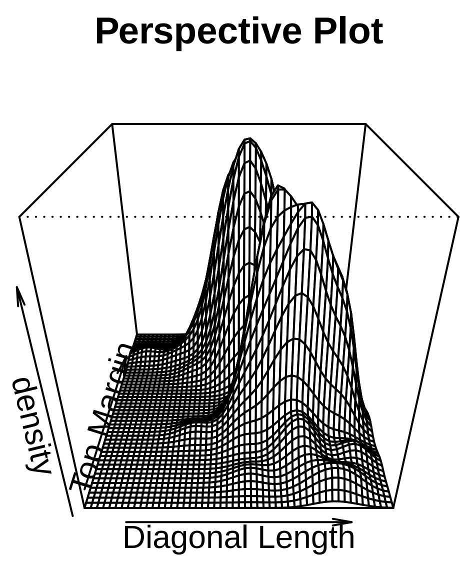
```
density <- bkde2D(a, bandwidth = bw)
plot(a,kernel = "gaussian",
     xlab = "Diagonal Length",
     ylab = "Top Margin",
     main = "Contour Plot")

contour(x = density$x1, y = density$x2, z = density$fhat, add = TRUE)
```



Perspective plot

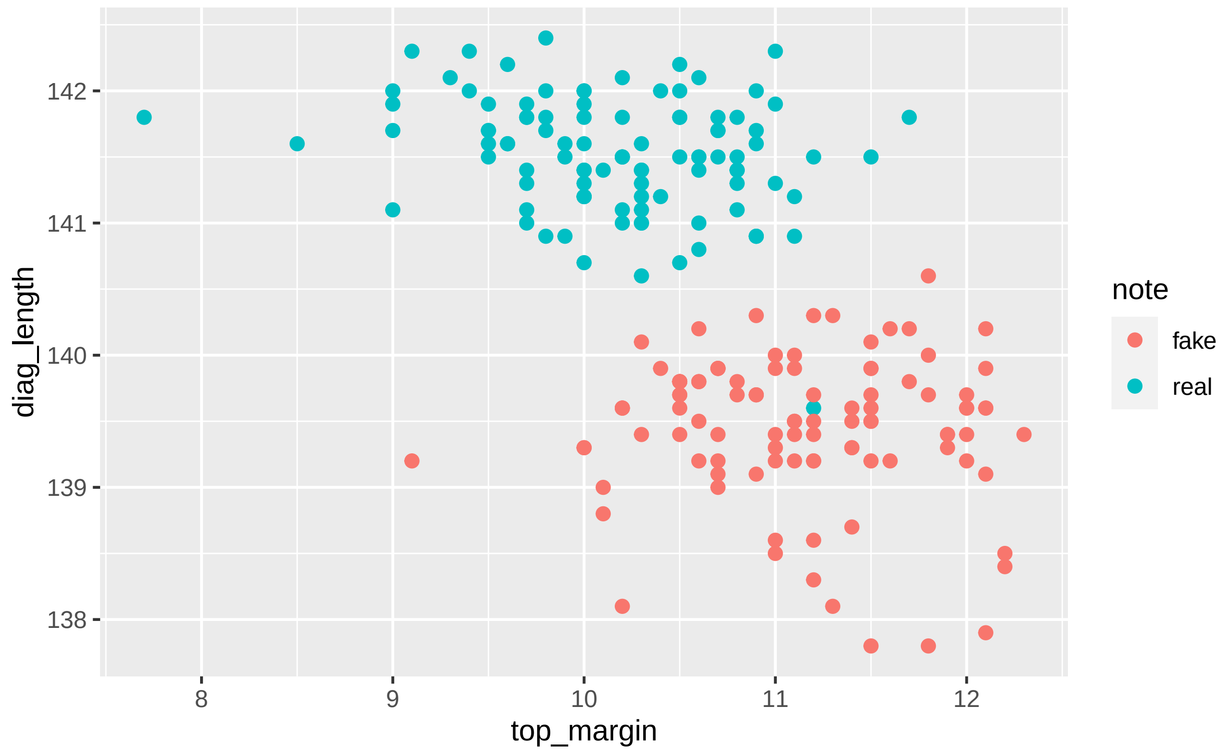
```
persp(x = density$x1, y = density$x2, z = density$fhat, phi=30, xlab = "Diagonal Length",
      ylab = "Top Margin",
      main = "Perspective Plot",
      zlab = "density"
)
```



c. Plot the scatterplot, highlighting the points with different colors according to whether the bills are real or fake. Explain your findings.

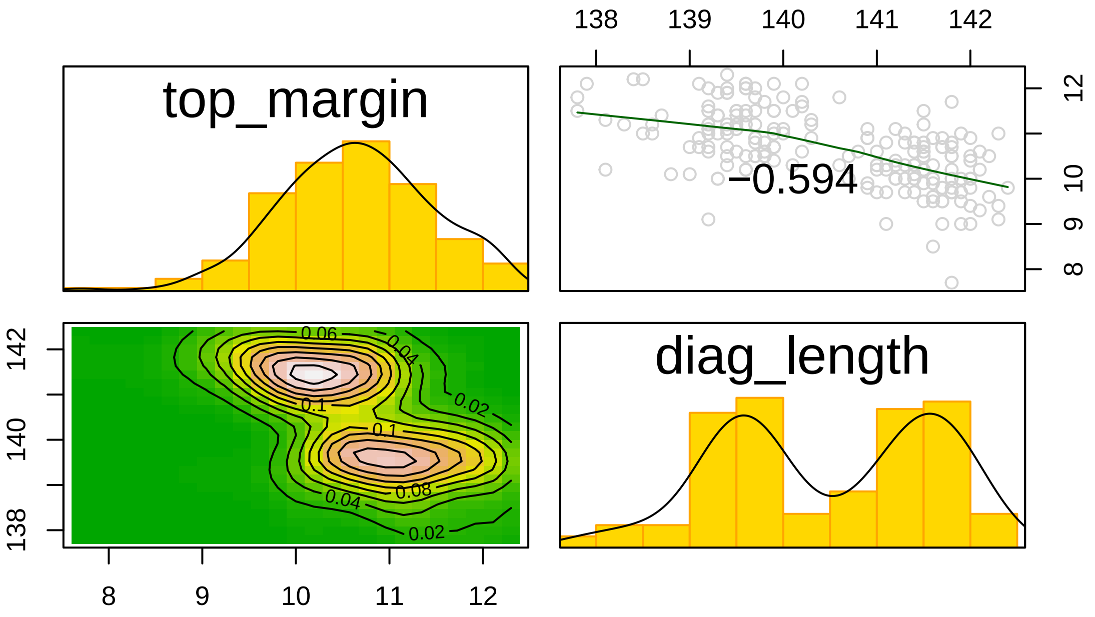
Scatterplot

```
library(ggplot2)
ggplot(subswiss, aes(x=top_margin, y=diag_length, color=note)) +
  geom_point(size=2)
```



Scatterplot matrix with density information

```
kdepairs(subswiss[,c("top_margin", "diag_length")])
```



Problem 9: Examine the multivariate normality of the BCG data from the HSAUR2 package

```
library(HSAUR2)
```

```
data(BCG, package = "HSAUR2")
BCG
```

```
##    Study BCGTB BCGVacc NoVaccTB NoVacc Latitude Year
## 1      1     4     123       11    139       44 1948
## 2      2     6     306       29    303       55 1949
## 3      3     3     231       11    220       42 1960
## 4      4    62   13598      248  12867       52 1977
## 5      5    33    5069       47   5808       13 1973
## 6      6   180    1541      372   1451       44 1953
## 7      7     8    2545       10    629       19 1973
## 8      8   505   88391      499  88391       13 1980
## 9      9    29    7499       45   7277       27 1968
## 10    10    17    1716       65   1665       42 1961
## 11    11   186   50634      141  27338       18 1974
## 12    12     5    2498        3   2341       33 1969
## 13    13    27   16913       29  17854       33 1976
```

excluding the year and the study number. Document your process as follows.

a. Find the column-means vector

First we will drop year and study number columns from BCG data.

```
BCG <- BCG[,-c(1,7) ]
BCG
```

```
##    BCGTB BCGVacc NoVaccTB NoVacc Latitude
## 1      4     123       11    139       44
## 2      6     306       29    303       55
## 3      3     231       11    220       42
## 4     62   13598      248  12867       52
## 5     33    5069       47   5808       13
## 6    180    1541      372   1451       44
## 7      8    2545       10    629       19
## 8    505   88391      499  88391       13
## 9     29    7499       45   7277       27
## 10    17    1716       65   1665       42
## 11   186   50634      141  27338       18
## 12     5    2498        3   2341       33
## 13    27   16913       29  17854       33
```

Column means vector

```
BCGbar <- colMeans(BCG)
BCGbar
```

```
##       BCGTB     BCGVacc    NoVaccTB      NoVacc    Latitude
##    81.92308 14697.23077   116.15385 12791.00000    33.46154
```

b. Find the covariance matrix

Covariance matrix

```
BCGcov <- cov(BCG)
BCGcov
```

```
##                  BCGTB     BCGVacc     NoVaccTB      NoVacc     Latitude
## BCGTB       20142.9103   3369143.9   20161.9295   3174306.1    -901.1282
## BCGVacc   3369143.8526 678311880.4 2851888.9615 610578756.8 -208049.6987
## NoVaccTB    20161.9295   2851889.0   25322.4744   2802552.2    -324.5769
## NoVacc    3174306.0833 610578756.8 2802552.1667 584611245.7 -178139.1667
## Latitude     -901.1282   -208049.7    -324.5769   -178139.2     208.6026
```

c. Find the Mahalanobis distances using the data and the values determined in parts (a) and (b)

Mahalanobis distances

```
d2 <- mahalanobis(BCG, BCGbar, BCGcov)
d2
```

```
##          1          2          3          4          5          6          7
##  1.7431300  4.1009635  1.3112740  9.6960849  4.6249148  9.6218066  2.5447445
##          8          9         10         11         12         13
## 10.8362249  0.7833235  0.4222457 10.9639632  0.7299988  2.6213256
```

d. Sort the Mahalanobis distances from smallest to largest

Sorted Mahalanobis distances

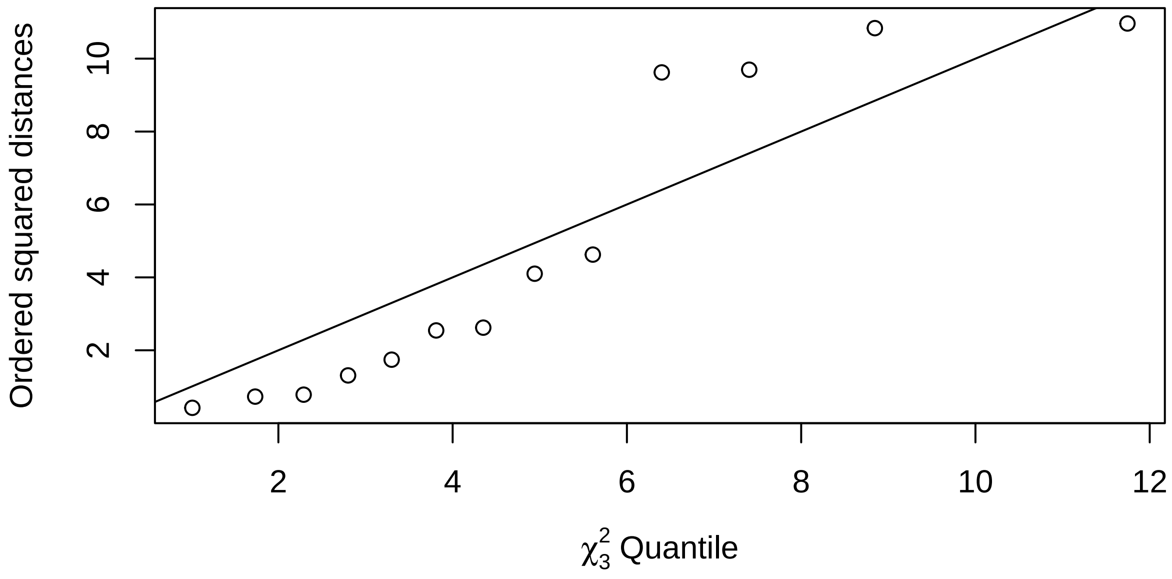
```
sort_d2=sort(d2, decreasing = F)
sort_d2
```

```
##         10         12          9          3          1          7         13
##  0.4222457  0.7299988  0.7833235  1.3112740  1.7431300  2.5447445  2.6213256
##          2          5          6          4          8         11
##  4.1009635  4.6249148  9.6218066  9.6960849 10.8362249 10.9639632
```

e. Plot the sorte d Mahalanobis distances vs the chi-square (of the appropriate number of degrees of freedom) quantiles.

Mahalanobis distances vs the chi-square Plot

```
quantiles <- qchisq((1:nrow(BCG) - 1/2)/nrow(BCG), df = ncol(BCG))
plot(quantiles, sort_d2,
xlab = expression(paste(chi[3]^2, " Quantile")),
ylab = "Ordered squared distances")
abline(a = 0, b = 1)
```



f. Interpret the data. Is it multivariate normal?

For data to be multivariate normal, its chi-squared distribution should lead to a straight line through the origin, and there should not be outliers present in distribution. As we can see in (e), the straight line has not originated from origin and outliers are also seen in distribution so the given data is approximately not a multivariate normal.