Lab1

Naveen (navga709), Sridhar(sriad858), Juan(juado206), Samia(sambu064)

2019-11-25

Contents

1. Describing individual variables

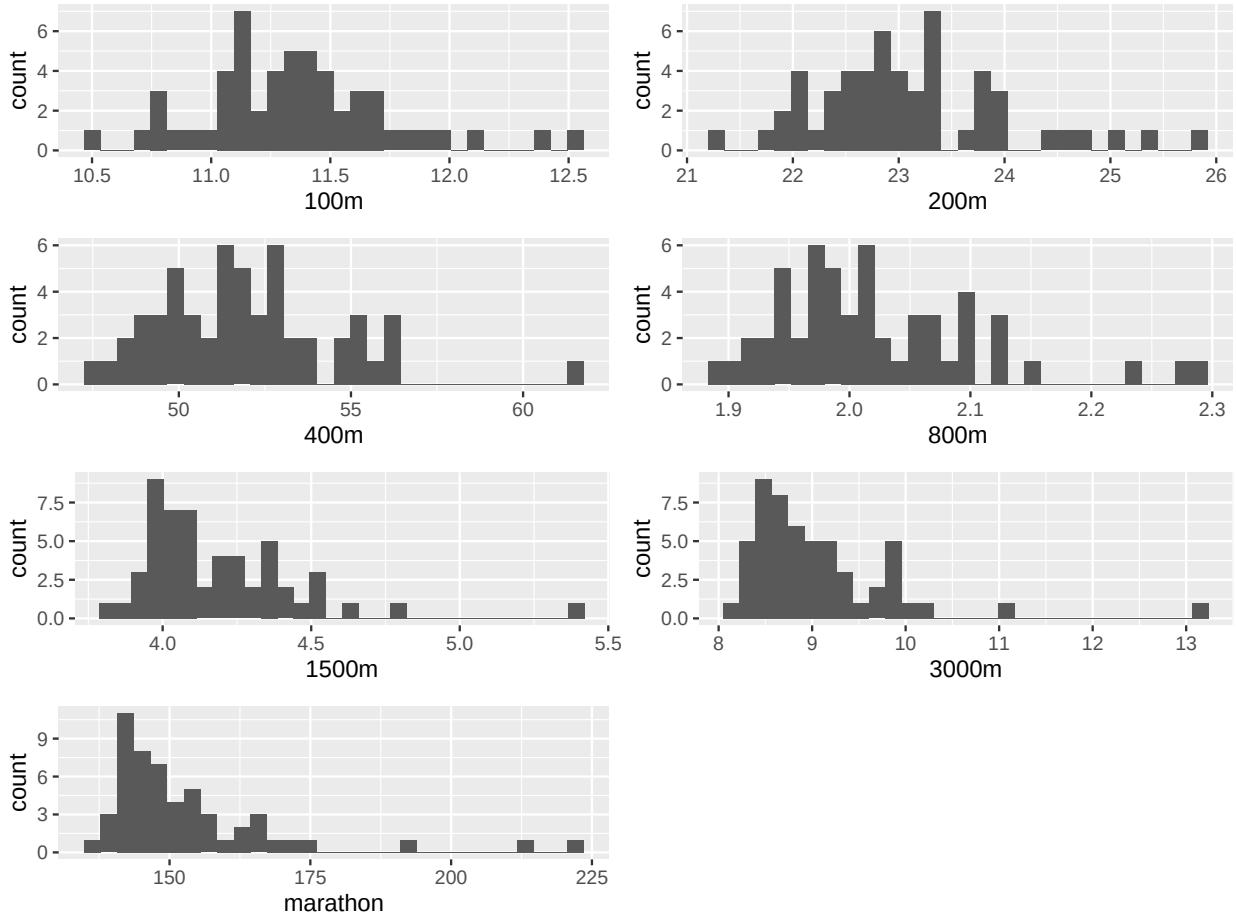
a - Describe the 7 variables with mean values, standard deviations e.t.c.

```
  Country  100m  200m  400m 800m 1500m 3000m marathon
1     ARG 11.57 22.94 52.50 2.05  4.25  9.19   150.32
2     AUS 11.12 22.23 48.63 1.98  4.02  8.63   143.51
3     AUT 11.15 22.70 50.62 1.94  4.05  8.78   154.35
4     BEL 11.14 22.48 51.45 1.97  4.08  8.82   143.05
5     BER 11.46 23.05 53.30 2.07  4.29  9.81   174.18
```

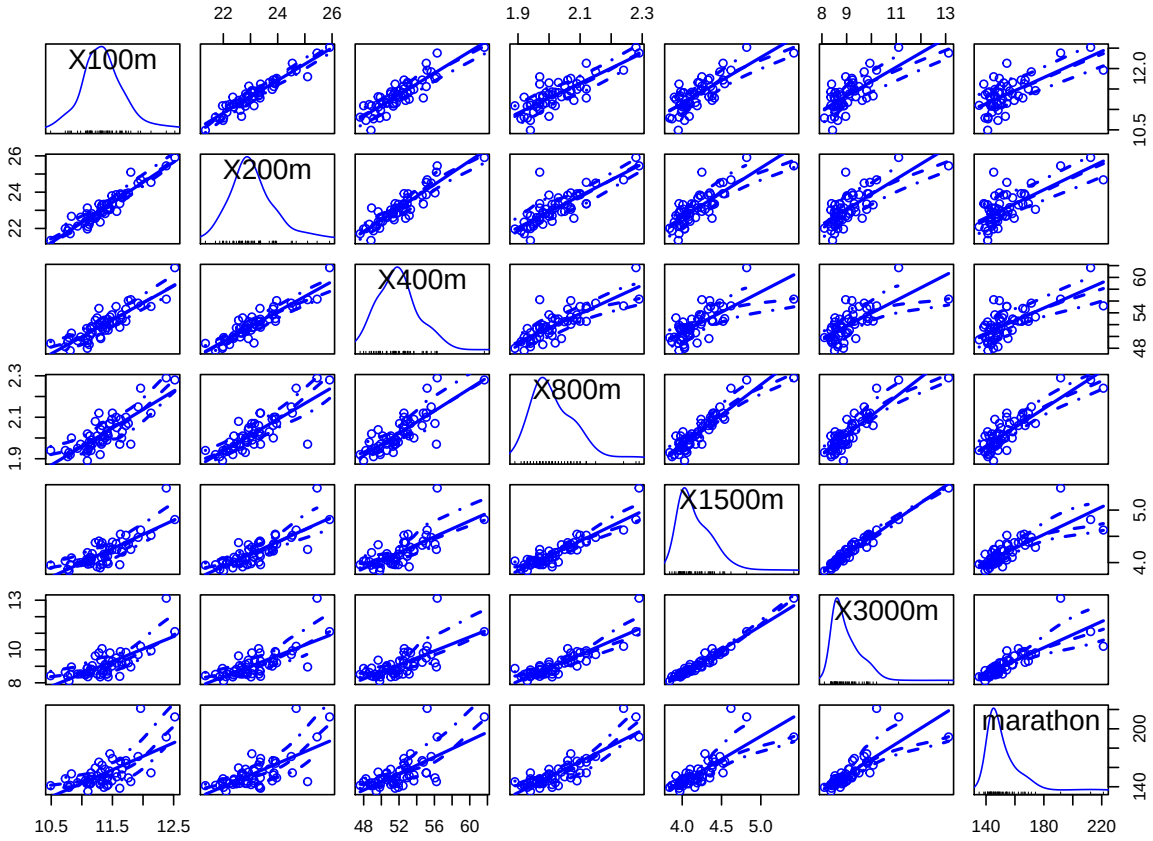
Statistic	Min	Pctl(25)	Median	Pctl(75)	Max	Median	St. Dev.
100m	10.490	11.123	11.325	11.568	12.520	11.325	0.394
200m	21.340	22.570	22.980	23.610	25.910	22.980	0.929
400m	47.600	49.968	51.645	53.117	61.650	51.645	2.597
800m	1.890	1.970	2.005	2.070	2.290	2.005	0.087
1500m	3.840	4.002	4.100	4.338	5.420	4.100	0.272
3000m	8.100	8.543	8.845	9.325	13.120	8.845	0.815
marathon	135.250	143.480	148.430	157.665	221.140	148.430	16.440

b - Illustrate the variables with different graphs (explore what plotting possibilities R has)

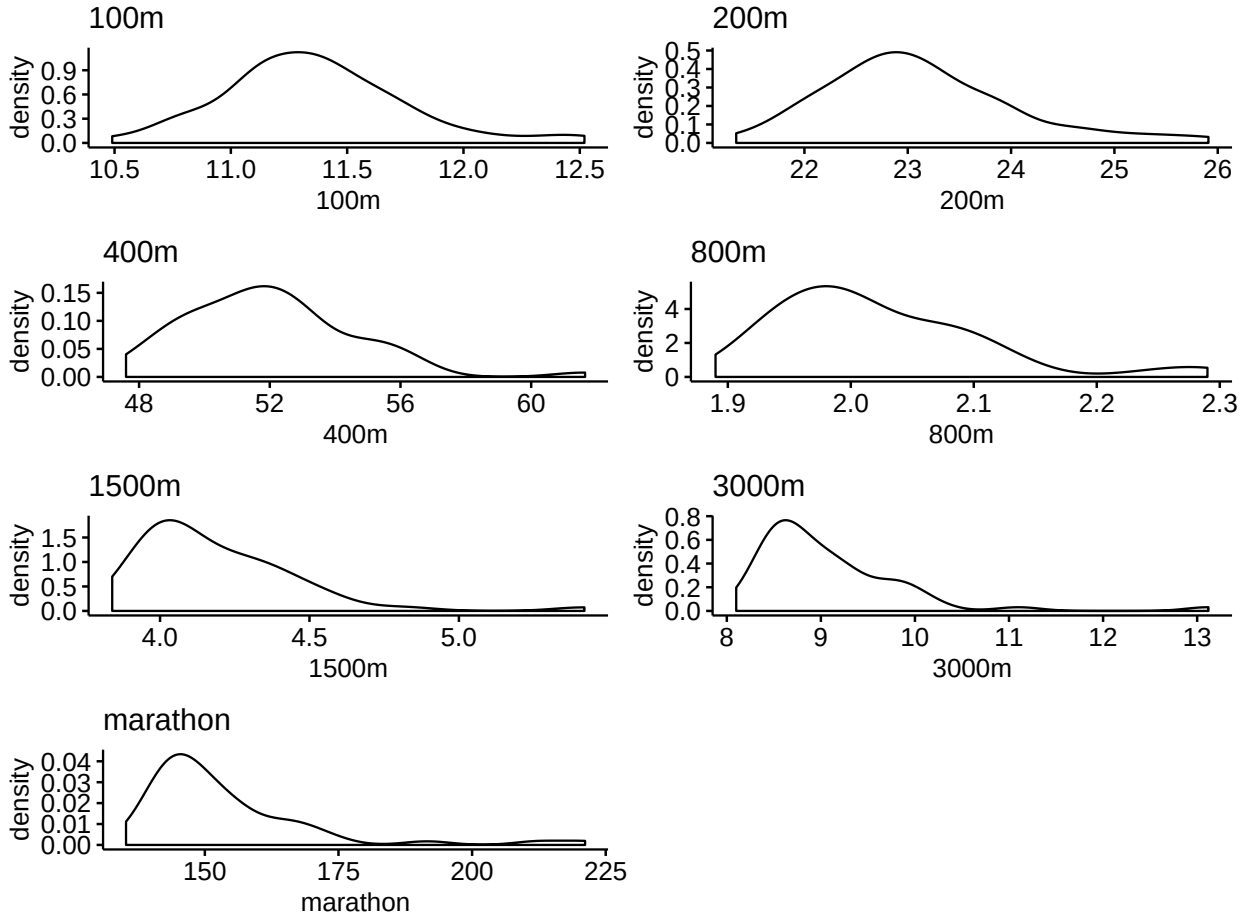
Below graph shows the summarized distribution of each variable



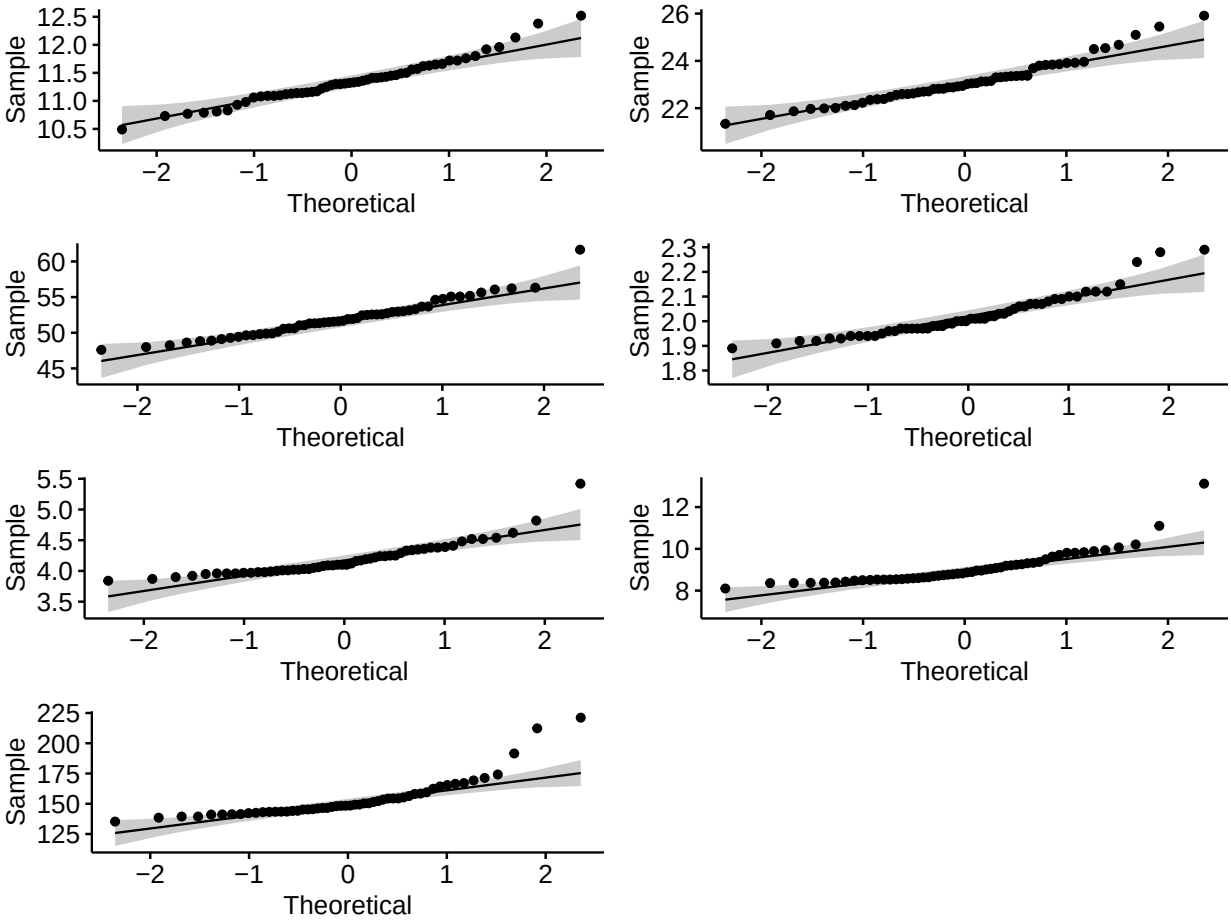
Below graph shows the relationship between variables.



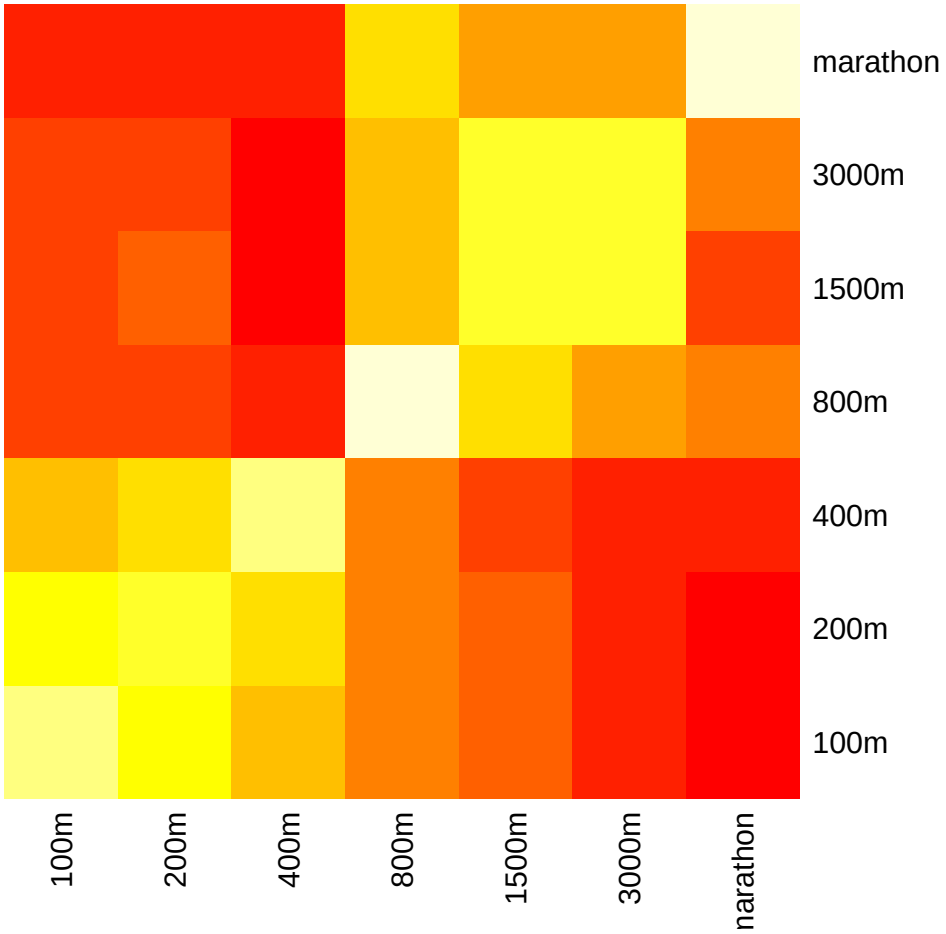
Below graph shows either variables are normally distributed or not.



Below graph shows the correlation between the features and the normal distribution



Below graphs is the color-coded graphical reporesentation of variables to better visualize the location of data.



2. Relationships between the variables

a - Compute the covariance and correlation matrices for the 7 variables. Is there any apparent structure in them? Save these matrices for future use.

For computing both matrices asked, it is enough to call the two functions associated with covariance and correlation, which are:

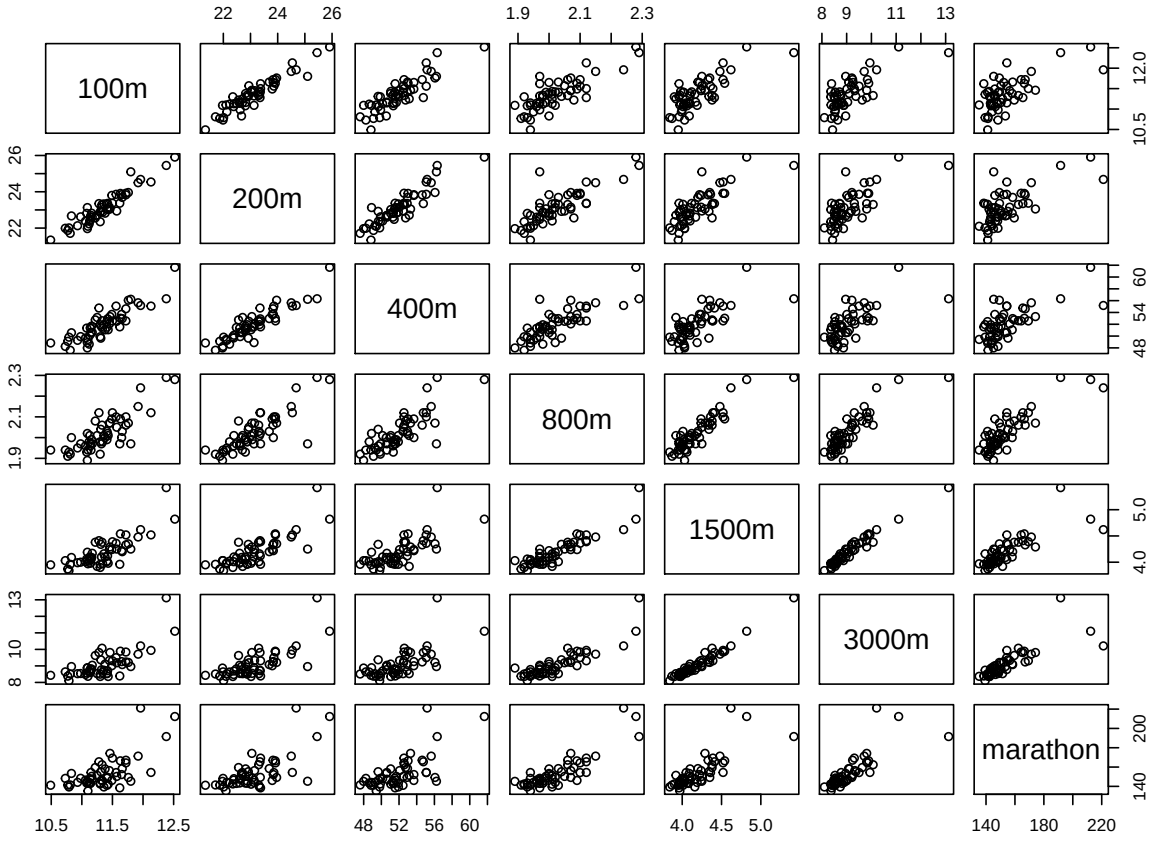
```
# 2a)
t1 <- data_T1
dataset = t1
cov_matrix <- cov(dataset[, 2:8])
cor_matrix <- cor(dataset[, 2:8])
```

Those results highlight the symmetry of both matrices. Furthermore, from this particular dataset shows how only looking to the covariance matrix a strong relationship between X8 and the rest of the variables could be inferred, but then the correlation matrix turns down the hypothesis.

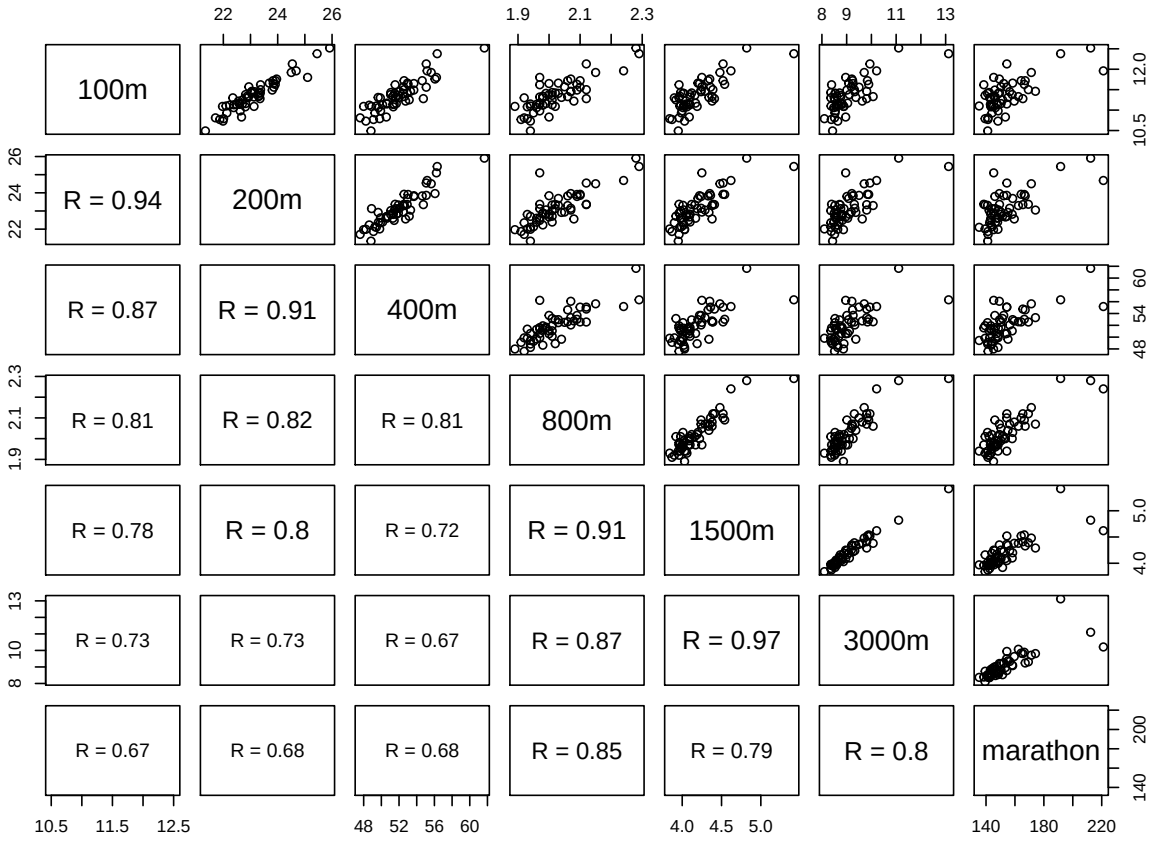
b - Generate and study the scatterplots between each pair of variables. Any extreme values?

These scatterplots can be generated by the following command:

```
# 2b)
pairs(dataset[, 2:8])
```

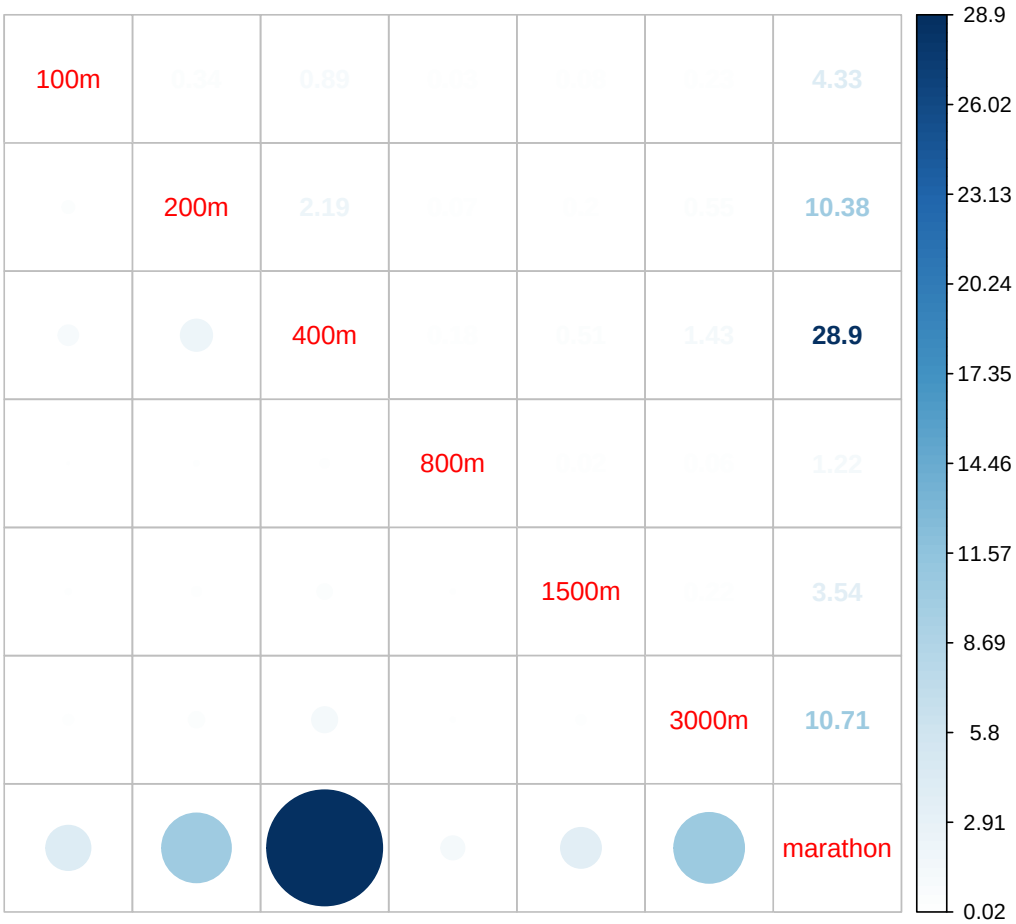


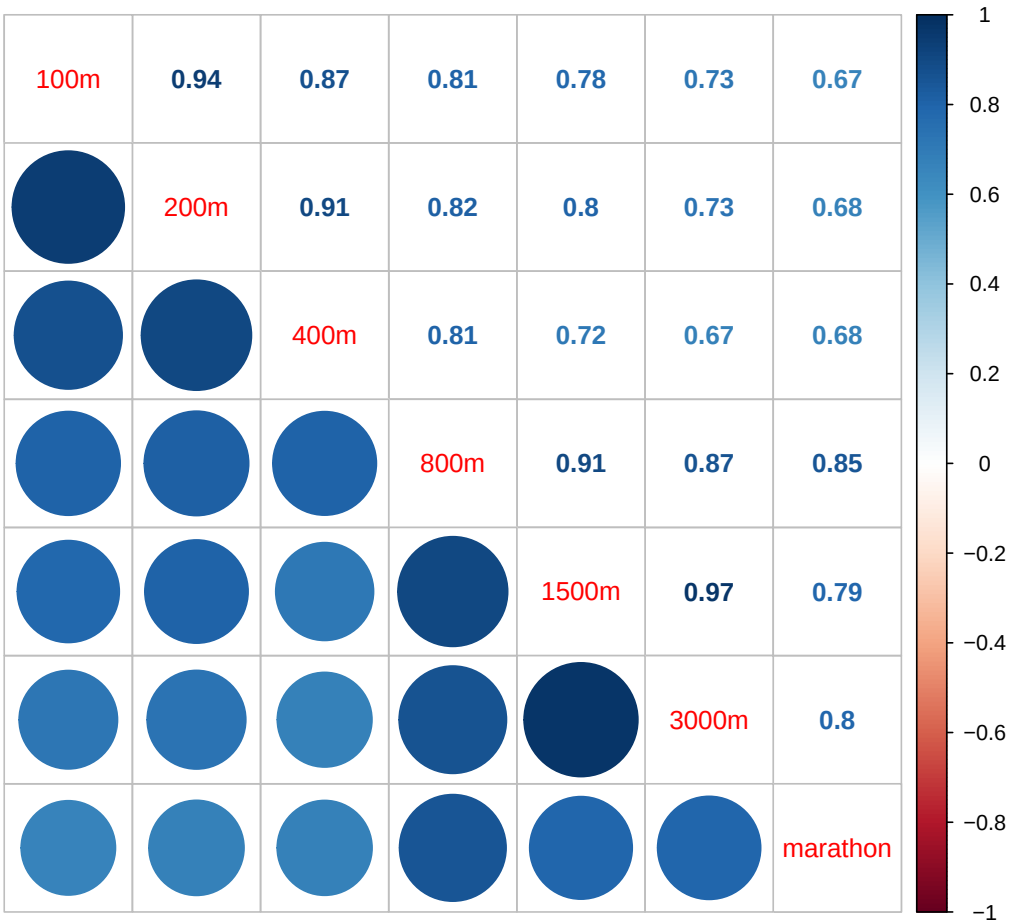
Deducing the strongest and weakest relationship from this plot could mean a hard job, but the arguments of the previous function can be slightly modified in order to facilitate the work. The panel.cor function calculates the correlation between two certain variables and associates a font size to it. The new figure looks like the following:



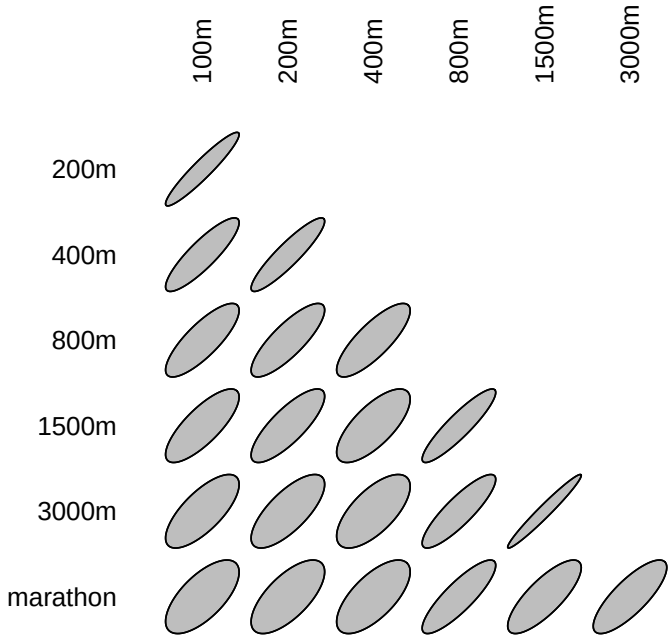
c - Explore what other plotting possibilities R offers for multivariate data. Present other (at least two) graphs that you find interesting with respect to this data set.

R offers interesting option for plotting a multivariate dataset. Including the needed libraries, three attractive figures are shown. If the following code is typed.


100m
200m
400m
800m
1500m
3000m
marathon
4.33
10.38
28.9
1.22
3.54
10.71
28.9
26.02
23.13
20.24
17.35
14.46
11.57
8.69
5.8
2.91
0.02


100m 0.94 0.87 0.81 0.78 0.73 0.67
200m 0.91 0.82 0.8 0.73 0.68
400m 0.81 0.72 0.67 0.68
800m 0.91 0.87 0.85
1500m 0.97 0.79
3000m 0.8
marathon
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

Bivariate correlations

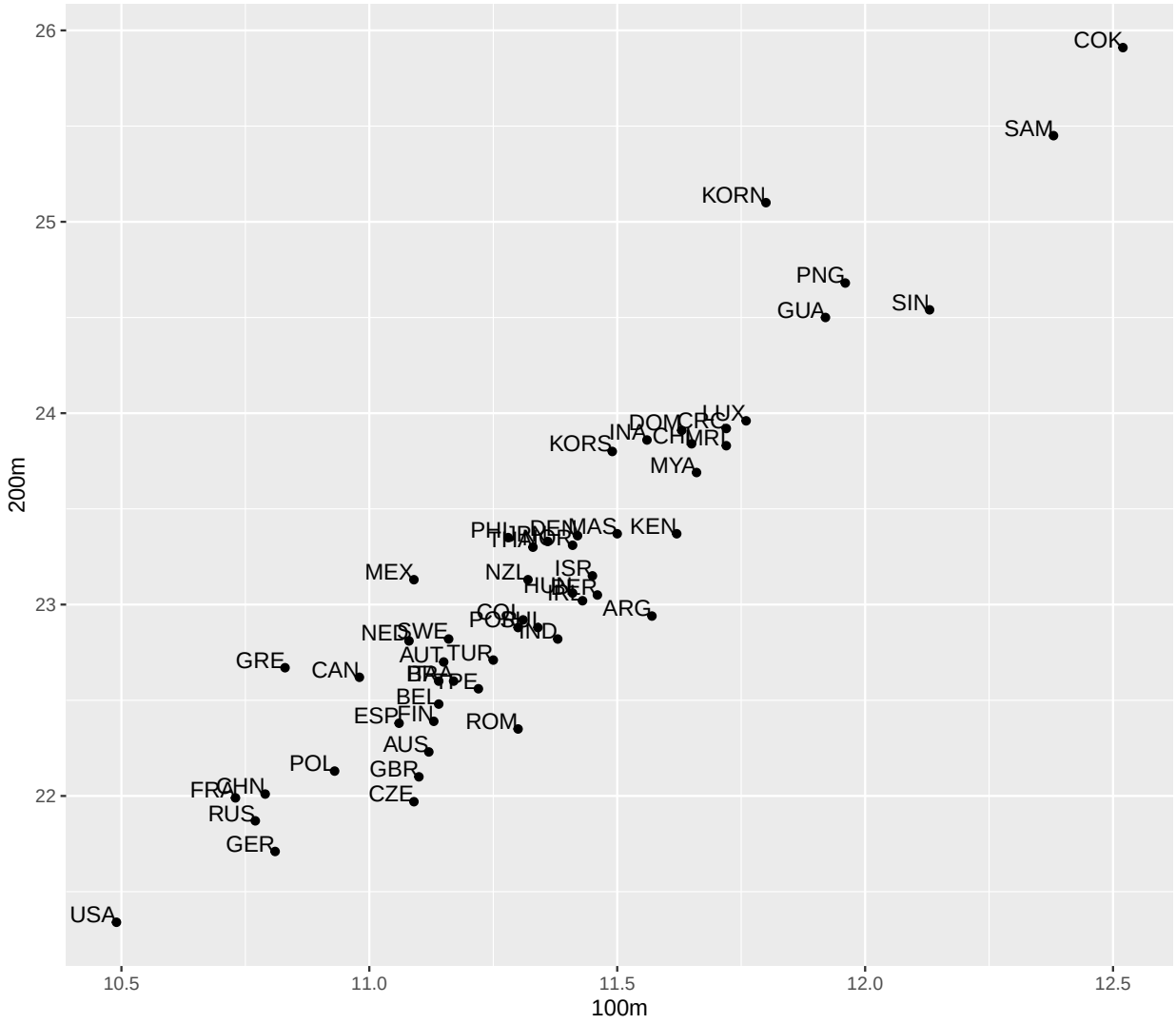


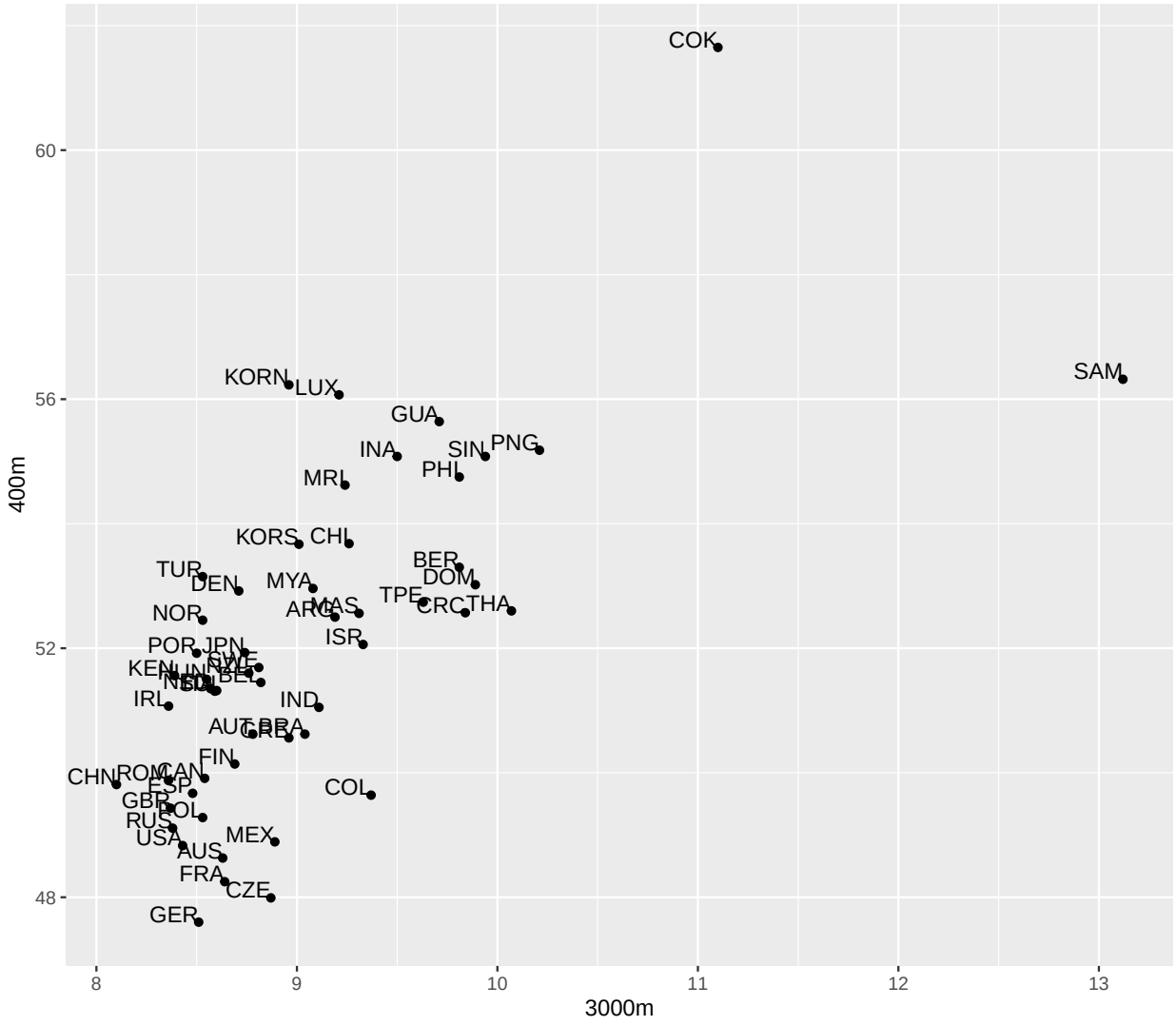
3. Examining for extreme values

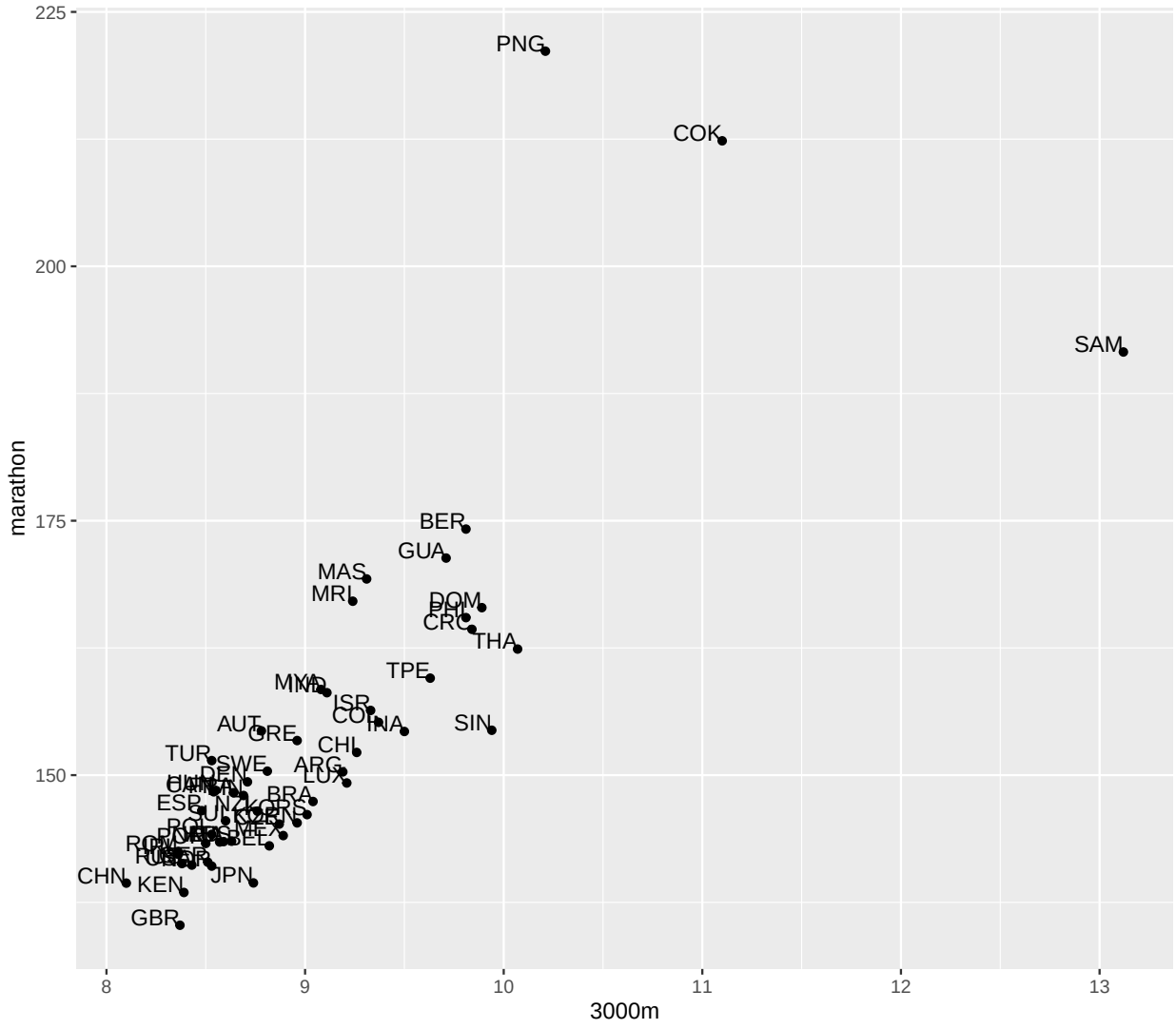
a - look for extreme values in plot

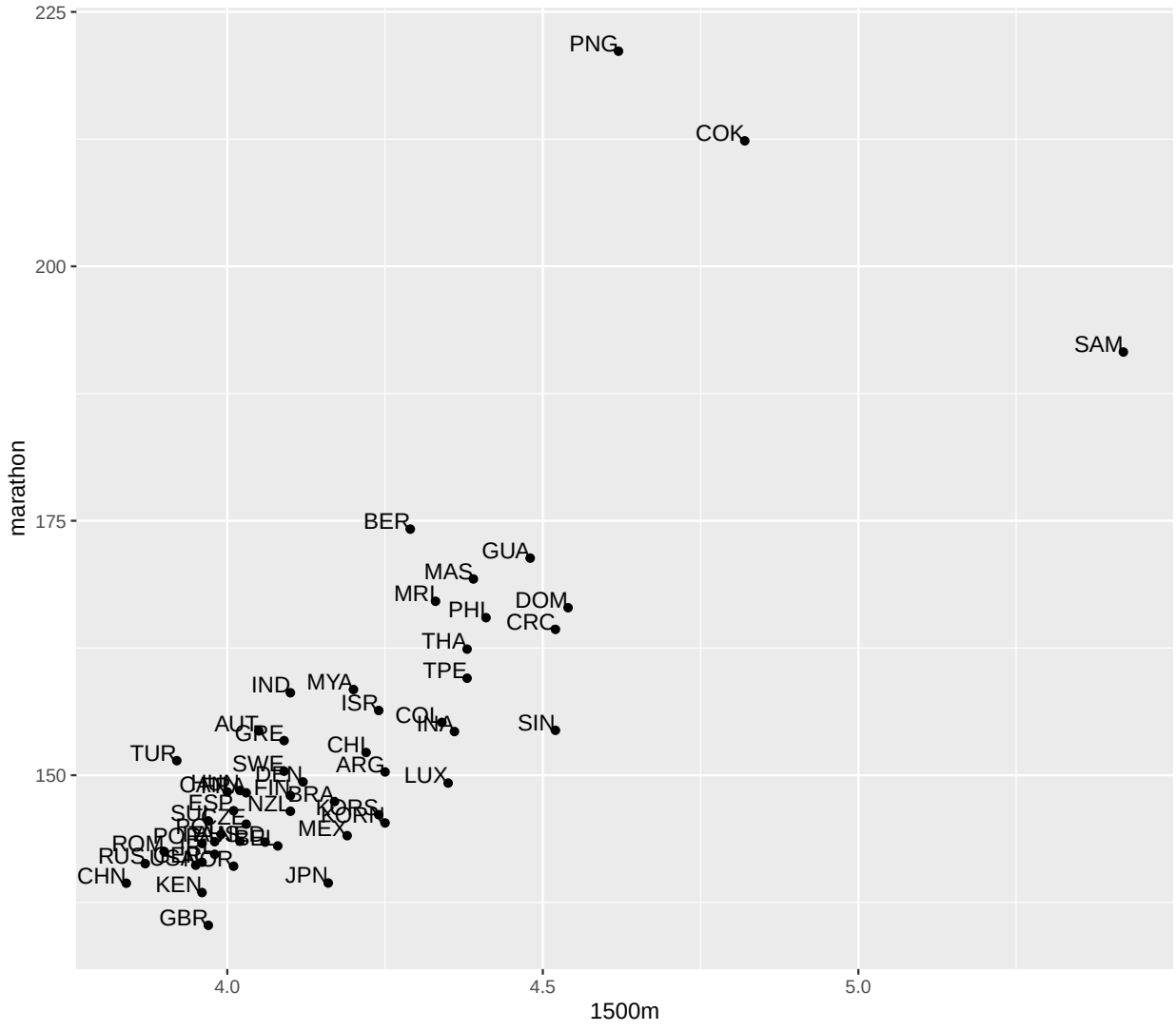
The countries COK, SAM and PNG are the three countries which appear most extreme to us. They perform poorly in all race patterns. The common thing we found about these countries is that all of them are small islands. Their values were always far away from the mean values in most of the race patterns.

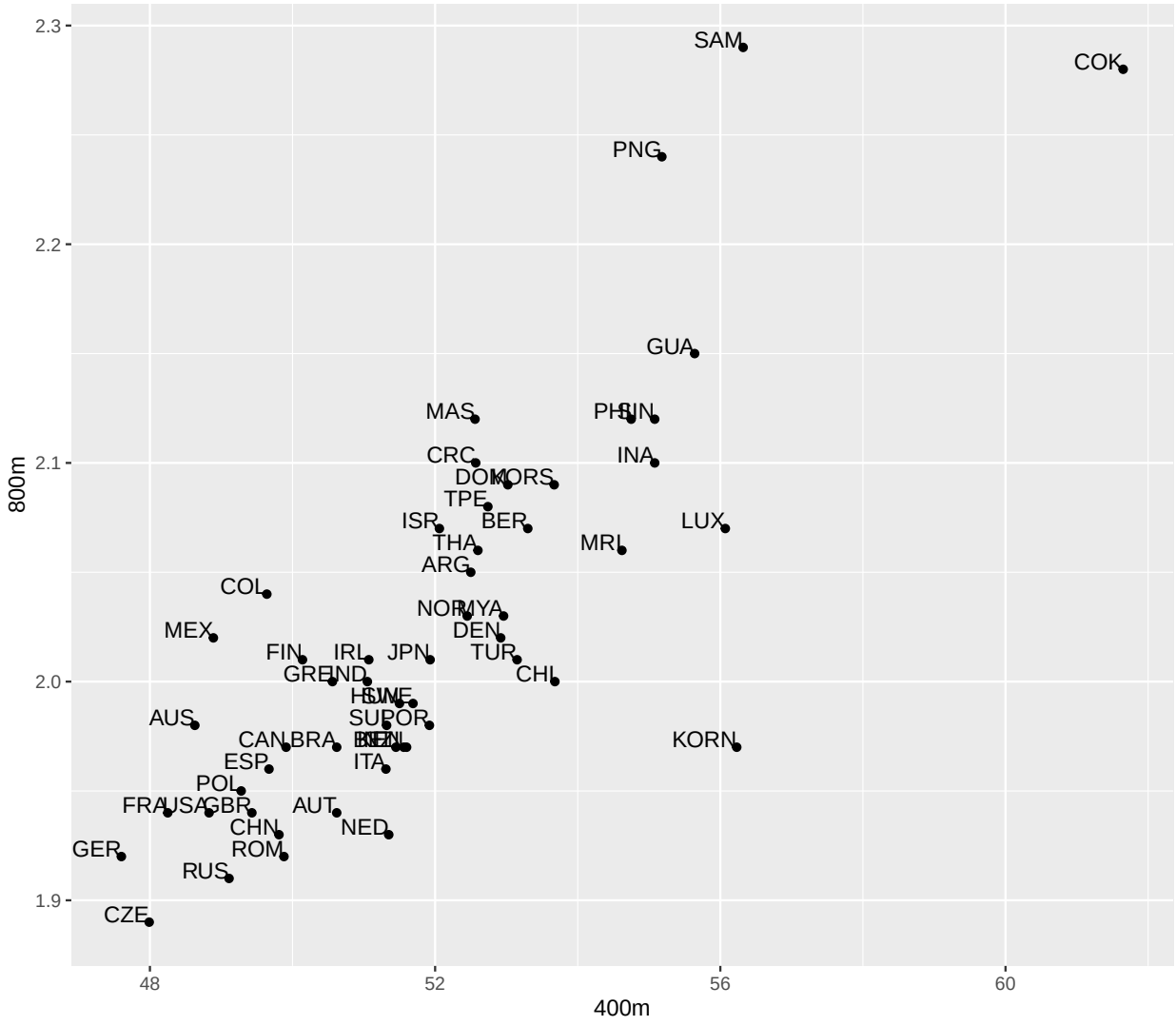
In the short distance races patterns, USA stands out but not in the long distance race pattens. This is the reason we are not considering USA as an extreme value for the entire dataset.

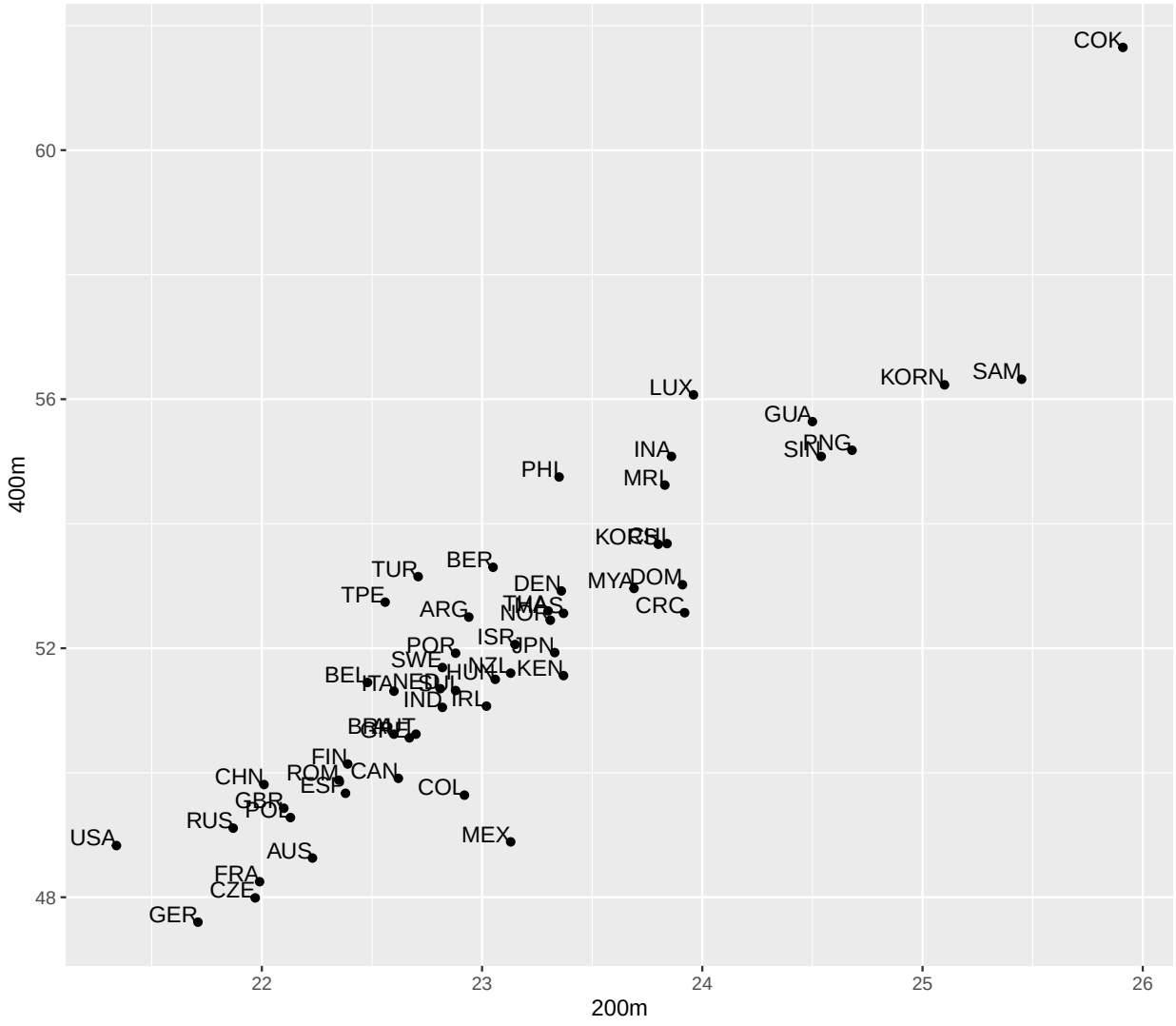



COK
SAM
KORN
LUX
GUA
INA
SIN
PNG
PHI
MRI
KORS
CHI
BER
DOM
TUR
DEN
MYA
TPE
CRC
THA
MAS
NOR
ISR
POR
JPN
IRL
IND
FIN
CHN
ESP
COL
GBR
RUS
USA
MEX
AUS
FRA
CZE
GER
60
56
52
48
400m
8
9
10
11
12
13
3000m


225
200
175
150
marathon
PNG
COK
SAM
BER
GUA
MAS
MRI
DOM
CRC
THA
TPE
ISR
AUT
GRE
CHI
SIN
TUR
SWE
ARG
LUX
BRA
ESP
CHN
KEN
JPN
GBR
8
9
10
11
12
13
3000m


225
200
175
150
marathon
PNG
COK
SAM
BER
GUA
MAS
MRI
PHI
DOM
CRC
THA
TPE
IND
MYA
ISR
COL
INA
SIN
AUT
GRE
CHI
TUR
SWE
ARG
LUX
FIN
BRA
ESP
NZL
MEX
CHN
KEN
JPN
GBR
4.0
4.5
5.0
1500m


800m
400m
2.3
2.2
2.1
2.0
1.9
48
52
56
60
SAM
COK
PNG
GUA
MAS
PHI
SIN
CRC
INA
DOM
KORS
TPE
ISR
BER
LUX
THA
MRI
ARG
COL
NOR
MYA
DEN
MEX
FIN
IRL
JPN
TUR
GRE
IND
CHI
AUS
SUI
POR
CAN
BRA
KORN
ESP
ITA
POL
FRA
USA
GBR
AUT
CHN
NED
GER
ROM
RUS
CZE



b - Euclidean Distance

```
squaredEucl = function(data, top_n=5, inv_mat=diag(1, 7, 7)){
  x_bar <- colMeans(data[,2:8])   #Taking means of column
  data_new<- apply(data[,2:8],1,function(x) (x-x_bar))  #Deviation from mean
  data_new <- as.matrix(t(data_new))       #Transposing to reflect original matrix
  dist <- sqrt(data_new %*% inv_mat %*% t(data_new))   #Calculating distance
  dist_mean <-  as.data.frame(diag(dist))     #Extracting distance from mean
  dist_mean$Countries <- data[,1]
  colnames(dist_mean)[1] <- "dist"
  dist_mean <- dist_mean[order(dist_mean$dist,decreasing = TRUE),]

  return(list(top_dis = dist_mean[1:top_n,], distMat = dist))
}

eq_dis = squaredEucl(t1, 5)
cat("\nVariance in the distances : ", var(diag(eq_dis$distMat)), "\n")
```

```
Variance in the distances :  144.3026
```

```
eq_dis$top_dis
```

```
       dist Countries
40 67.62796       PNG
11 59.61517       COK
46 38.52476       SAM
5  20.61606       BER
19 18.59146       GBR
```

Since the scale of the data varies a lot in this dataset, the variance in the distance measure is very high.

c - Scale independent euclidean distance.

```
scaleIndSquaredEucl = function(data, top_n=5){
  vari <-diag(apply(data[,2:8],2,var))
  inv_vari = solve(vari)

  return(squaredEucl(data, top_n = top_n, inv_mat = inv_vari))
}

siEq_dis = scaleIndSquaredEucl(t1, 5)
cat("\nVariance in the distances : ", var(diag(siEq_dis$distMat)), "\n")
```

```
Variance in the distances :  2.386019
```

```
siEq_dis$top_dis
```

```
       dist Countries
46 8.693837       SAM
11 8.037485       COK
40 5.850548       PNG
54 3.588439       USA
47 3.383026       SIN
```

On scaling the data with the sample variance of the variable the variance of the distance decreases a lot. The scale independent euclidean distance has the different order than the previous one, but the top three countries remain the same. The distance measure has decreased considerably.

d - Mahalanobis Distance

```
mahalanobisDist = function(data, top_n=5){
  inv_cov = solve(cov(data[2:8]))
  return(squaredEucl(data, top_n = top_n, inv_mat = inv_cov))
}

mah_dis = mahalanobisDist(t1, 5)
cat("\nVariance in the distances : ", var(diag(mah_dis$distMat)), "\n")
```

```
Variance in the distances :  1.117324
```

```
mah_dis$top_dis
```

```
       dist Countries
46 5.917268       SAM
40 5.523337       PNG
31 5.115383      KORN
11 4.453538       COK
35 3.772391       MEX
```

The variance of the distance decreases even further using Mahalanobis distance measure.

e - Summary

The Euclidean distance just takes the squared deviation from the mean of a variable into account when calculating the distance. This leads to a circular decision boundry. The Mahalanobis distance takes the covariance into accound, which leads to elliptic decision boundaries.

In our data, different variables have different scales. This leads to the problem of one variable dominating the distance for that observation. Euclidean distance does not take care of this. This is the reason the distance measures calculated were so large, with the maximum distance being 67,62.

Where as in case of Mahalanobis distance we multiply with the inverse of the covariance matrix which acts like scaling of the data and makes the distance measurement scale independent.

The resultant extreme countries were pretty similar in all the distance measures. The countries "SAM", "PNG", "COK" were in the top 5 countries for all the distance measures, but their position in the top 5 changed.

```
cat("\nNumber of values smaller than zero : ", sum(squaredEucl(t1, 5)$dist<=0, na.rm = T), "\n")
```

```
Number of values smaller than zero :  0
```

```
#czMat = czek_matrix(squaredEucl(t1, 5)$dist)
print('Error in if (!all(x >= 0)) stop("Negative distances not supported!") : missing value where TRUE/
```

```
[1] "Error in if (!all(x >= 0)) stop(\"Negative distances not supported!\") : missing value where TRUE/
```

```
cat("\nNumber of values smaller than zero : ", sum(scaleIndSquaredEucl(t1, 5)$dist<=0, na.rm = T), "\n"
```

```
Number of values smaller than zero :  0
```

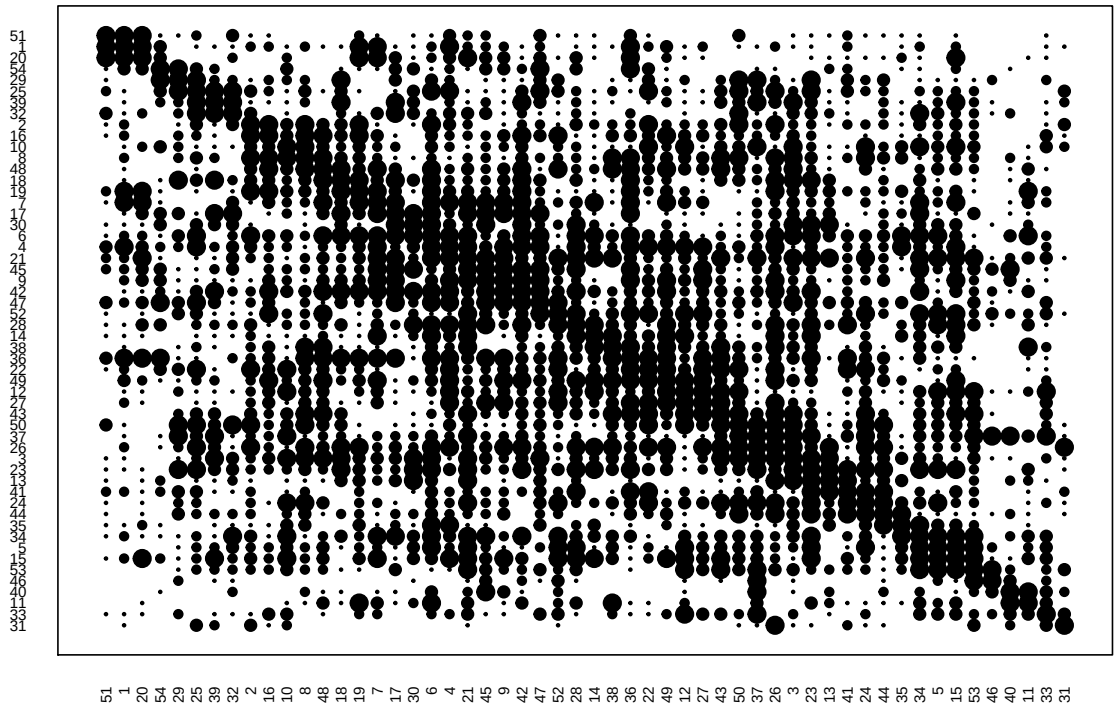
```
#czMat = czek_matrix(scaleIndSquaredEucl(t1, 5)$dist)
print('Error in if (!all(x >= 0)) stop("Negative distances not supported!") : missing value where TRUE/
```

```
[1] "Error in if (!all(x >= 0)) stop(\"Negative distances not supported!\") : missing value where TRUE/
```

When using this function ("czek_matrix") to convert a distance matrix (from Squared equalidian distance and Scale independent squared euclidean distance) into a Czekanowski matrix we were getting an error saying, "negative distances not supported". We checked if we and any values smaller than equal to zero but could not find any.

The reason for this could be that we are multiplying with a diagonal matrix in the first two cases, but in the case of mahalanobis distance we mumtiply with the inverse of the coavriance matrix.

```
czMat = czek_matrix(mahalanobisDist(t1, 5)$dist)
plot(czMat, values=c(1.5,1,0.75,0.25,0 ),
     plot_title = "Czekanowski's Diagram of Mahalanobis Distance matrix")
```



The function worked properly for the distance matrix from the Mahalanobis distance method.

Appendix

```
knitr::opts_chunk$set(
    message = FALSE,
    warning = FALSE,
    comment = NA,
    fig.width=8,
    fig.height=6
)
library(reshape)
library(kableExtra)
library(ggplot2)
library(ggpubr)
library(dplyr)
library(gridExtra)
library(stargazer)
library(gplots)
library(RMaCzek)
library(car)
data_T1 = read.csv("T1-9.dat", sep = "\t", header = F)
colnames(data_T1) = c("Country", "100m", "200m", "400m", "800m", "1500m", "3000m", "marathon")

data_T1 %>%
  head(5)

data_T1 %>%
  select(-Country) %>%
  stargazer(type = "text",
    summary.stat = c("min", "p25", "median", "p75", "max", "median", "sd")
)
`100m` =data_T1[,2]
`200m` =data_T1[,3]
`400m` =data_T1[,4]
`800m` =data_T1[,5]
`1500m` =data_T1[,6]
`3000m` =data_T1[,7]
marathon =data_T1[,8]

hp1<-ggplot(data_T1)+geom_histogram(aes(x=`100m`))
hp2<-ggplot(data_T1)+geom_histogram(aes(x=`200m`))
hp3<-ggplot(data_T1)+geom_histogram(aes(x=`400m`))
hp4<-ggplot(data_T1)+geom_histogram(aes(x=`800m`))
hp5<-ggplot(data_T1)+geom_histogram(aes(x=`1500m`))
hp6<-ggplot(data_T1)+geom_histogram(aes(x=`3000m`))
hp7<-ggplot(data_T1)+geom_histogram(aes(x=marathon))

grid.arrange(hp1,hp2,hp3,hp4,hp5,hp6,hp7,ncol = 2)
scatterplotMatrix(data_T1[,-1])
p01<-ggdensity(data_T1[,2],
          main = names(data_T1)[2],
          xlab = names(data_T1)[2])

p02<-ggdensity(data_T1[,3],
```

```
          main = names(data_T1)[3],
          xlab = names(data_T1)[3])

p03<-ggdensity(data_T1[,4],
          main = names(data_T1)[4],
          xlab = names(data_T1)[4])

p04<-ggdensity(data_T1[,5],
          main = names(data_T1)[5],
          xlab = names(data_T1)[5])

p05<-ggdensity(data_T1[,6],
          main = names(data_T1)[6],
          xlab = names(data_T1)[6])

p06<-ggdensity(data_T1[,7],
          main = names(data_T1)[7],
          xlab = names(data_T1)[7])

p07<-ggdensity(data_T1[,8],
          main = names(data_T1)[8],
          xlab = names(data_T1)[8])

grid.arrange(p01,p02,p03,p04,p05,p06,p07,ncol = 2)
gq1<-ggqqplot(data_T1[,2])
gq2<-ggqqplot(data_T1[,3])
gq3<-ggqqplot(data_T1[,4])
gq4<-ggqqplot(data_T1[,5])
gq5<-ggqqplot(data_T1[,6])
gq6<-ggqqplot(data_T1[,7])
gq7<-ggqqplot(data_T1[,8])
grid.arrange(gq1,gq2,gq3,gq4,gq5,gq6,gq7,ncol = 2)
heatmap(cor(data_T1[,-1]),Rowv=NA,Colv=NA)
# 2a)
t1 <- data_T1
dataset = t1
cov_matrix <- cov(dataset[, 2:8])
cor_matrix <- cor(dataset[, 2:8])
# 2b)
pairs(dataset[, 2:8])
panel.cor <- function(x, y){
  usr <- par("usr"); on.exit(par(usr))
  par(usr = c(0, 1, 0, 1))
  r <- round(cor(x, y), digits=2)
  txt <- paste0("R = ", r)
  cex.cor <- 0.8/strwidth(txt)
  text(0.5, 0.5, txt, cex = cex.cor * r)
}

pairs(dataset[, 2:8], lower.panel = panel.cor)
# 2c)
library(corrplot)
library(ellipse)
```

```
corrplot.mixed(cov_matrix, is.corr = FALSE, upper = "number", lower = "circle")
corrplot.mixed(cor_matrix, upper = "number", lower = "circle")
plotcorr(cor_matrix, type="lower", diag=FALSE, main="Bivariate correlations")
t1 <- as.data.frame(t1)

ggplot(t1,aes(x=`100m`,y=`200m`)) + geom_point() +geom_text(aes(label=Country,hjust=1,vjust=0))

ggplot(t1,aes(x=`3000m`,y=`400m`)) + geom_point() +geom_text(aes(label=Country,hjust=1,vjust=0))

ggplot(t1,aes(x=`3000m`,y=marathon)) + geom_point() +geom_text(aes(label=Country,hjust=1,vjust=0))

ggplot(t1,aes(x=`1500m`,y=marathon)) + geom_point() +geom_text(aes(label=Country,hjust=1,vjust=0))

ggplot(t1,aes(x=`400m`,y=`800m`)) + geom_point() +geom_text(aes(label=Country,hjust=1,vjust=0))

ggplot(t1,aes(x=`200m`,y=`400m`)) + geom_point() +geom_text(aes(label=Country,hjust=1,vjust=0))

squaredEucl = function(data, top_n=5, inv_mat=diag(1, 7, 7)){
  x_bar <- colMeans(data[,2:8])   #Taking means of column
  data_new<- apply(data[,2:8],1,function(x) (x-x_bar))  #Deviation from mean
  data_new <- as.matrix(t(data_new))       #Transposing to reflect original matrix
  dist <- sqrt(data_new %*% inv_mat %*% t(data_new))   #Calculating distance
  dist_mean <-  as.data.frame(diag(dist))     #Extracting distance from mean
  dist_mean$Countries <- data[,1]
  colnames(dist_mean)[1] <- "dist"
  dist_mean <- dist_mean[order(dist_mean$dist,decreasing = TRUE),]

  return(list(top_dis = dist_mean[1:top_n,], distMat = dist))
}

eq_dis = squaredEucl(t1, 5)
cat("\nVariance in the distances : ", var(diag(eq_dis$distMat)), "\n")
eq_dis$top_dis
scaleIndSquaredEucl = function(data, top_n=5){
  vari <-diag(apply(data[,2:8],2,var))
  inv_vari = solve(vari)

  return(squaredEucl(data, top_n = top_n, inv_mat = inv_vari))
}

siEq_dis = scaleIndSquaredEucl(t1, 5)
cat("\nVariance in the distances : ", var(diag(siEq_dis$distMat)), "\n")
siEq_dis$top_dis
mahalanobisDist = function(data, top_n=5){
  inv_cov = solve(cov(data[2:8]))
  return(squaredEucl(data, top_n = top_n, inv_mat = inv_cov))
}

mah_dis = mahalanobisDist(t1, 5)
cat("\nVariance in the distances : ", var(diag(mah_dis$distMat)), "\n")
mah_dis$top_dis
cat("\nNumber of values smaller than zero : ", sum(squaredEucl(t1, 5)$dist<=0, na.rm = T), "\n")
#czMat = czek_matrix(squaredEucl(t1, 5)$dist)
```

```
print('Error in if (!all(x >= 0)) stop("Negative distances not supported!") : missing value where TRUE/
cat("\nNumber of values smaller than zero : ", sum(scaleIndSquaredEucl(t1, 5)$dist<=0, na.rm = T), "\n"
#czMat = czek_matrix(scaleIndSquaredEucl(t1, 5)$dist)
print('Error in if (!all(x >= 0)) stop("Negative distances not supported!") : missing value where TRUE/
czMat = czek_matrix(mahalanobisDist(t1, 5)$dist)
plot(czMat, values=c(1.5,1,0.75,0.25,0 ),
     plot_title = "Czekanowski's Diagram of Mahalanobis Distance matrix")
```