Multivariate Statistical Methods (732A97) - Lab 01

Hector Plata (hecpl268)

Contents

Question 1: Describing individual variables

Consider the data set in the `T1-9.dat` file, National track records for women. For 55 different countries we heave the national records for 7 variables (100, 200, 400, 800, 1500, 3000m and marathon). Use `R` to do the following analyses.

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

```
data = read.table("T1-9.dat")
features = c("Country", "100", "200", "400", "800", "1500", "3000m", "Marathon")
colnames(data) = features

print(head(data))
```

```
##   Country   100   200   400  800 1500 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
## 6     BRA 11.17 22.60 50.62 1.97 4.17  9.04   147.41
```

```
print(summary(data))
```

```
##     Country        100             200             400       
##  ARG    : 1   Min.   :10.49   Min.   :21.34   Min.   :47.60  
##  AUS    : 1   1st Qu.:11.12   1st Qu.:22.57   1st Qu.:49.97  
##  AUT    : 1   Median :11.32   Median :22.98   Median :51.65  
##  BEL    : 1   Mean   :11.36   Mean   :23.12   Mean   :51.99  
##  BER    : 1   3rd Qu.:11.57   3rd Qu.:23.61   3rd Qu.:53.12  
##  BRA    : 1   Max.   :12.52   Max.   :25.91   Max.   :61.65  
##  (Other):48                                                  
##       800             1500           3000m           Marathon     
##  Min.   :1.890   Min.   :3.840   Min.   : 8.100   Min.   :135.2  
##  1st Qu.:1.970   1st Qu.:4.003   1st Qu.: 8.543   1st Qu.:143.5  
##  Median :2.005   Median :4.100   Median : 8.845   Median :148.4  
##  Mean   :2.022   Mean   :4.189   Mean   : 9.081   Mean   :153.6  
##  3rd Qu.:2.070   3rd Qu.:4.338   3rd Qu.: 9.325   3rd Qu.:157.7  
##  Max.   :2.290   Max.   :5.420   Max.   :13.120   Max.   :221.1  
##
```

```r
print("Standard deviations")
```

```
## [1] "Standard deviations"
```

```r
print(apply(data[2:8], 2, sd))
```

```
##         100         200         400         800        1500       3000m
##  0.39410116  0.92902547  2.59720188  0.08687304  0.27236502  0.81532689
##    Marathon
## 16.43989508
```

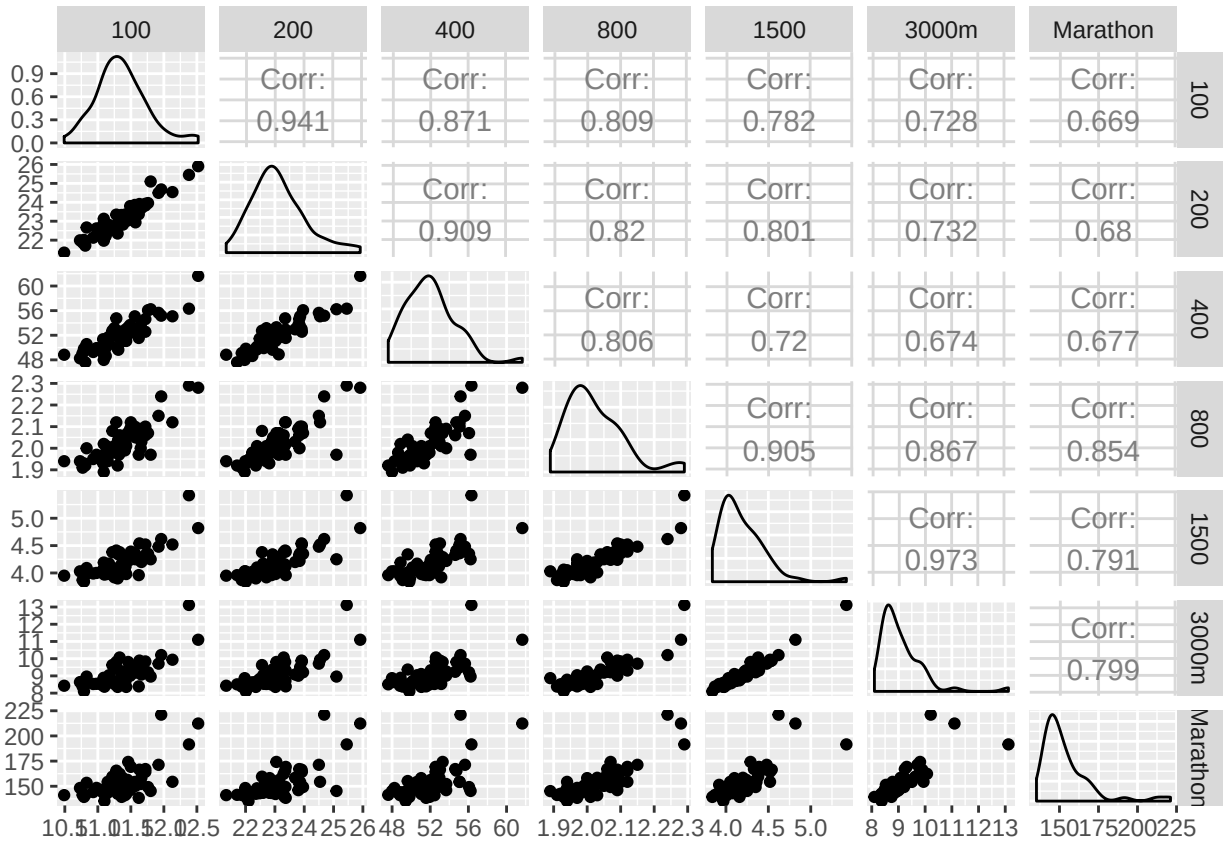
b) Illustrate the variables with different graphs (explore what plotting possibilities `R` has). Make sure that the graphs look attractive (it is absolutely necessary to look at the labels, font sizes, point types). Are there any apparent extreme values? Do the variables seem normally distributed? Plot the best fitting (match the mean and standard deviation, i.e. method of moments) Gaussian density curve on the data's histogram. For the last part you may be interested in `hist()` and `density()` functions.

For all of the variables there seems to be outliers towards their upper quantiles. The variables from their KDE on the diagonal, doesn't seem to follow a normal distribution. The closest variable to a normal distribution is the variable `100`. The quantile plots compare the quantiles of the data with a corresponding normal distribution. This confirms the outliers and the fact that they don't follow a normal distribution. Another issue with the data is that is truncated and thus by construction doesn't have the same domain of a normal distribution over the random variable.

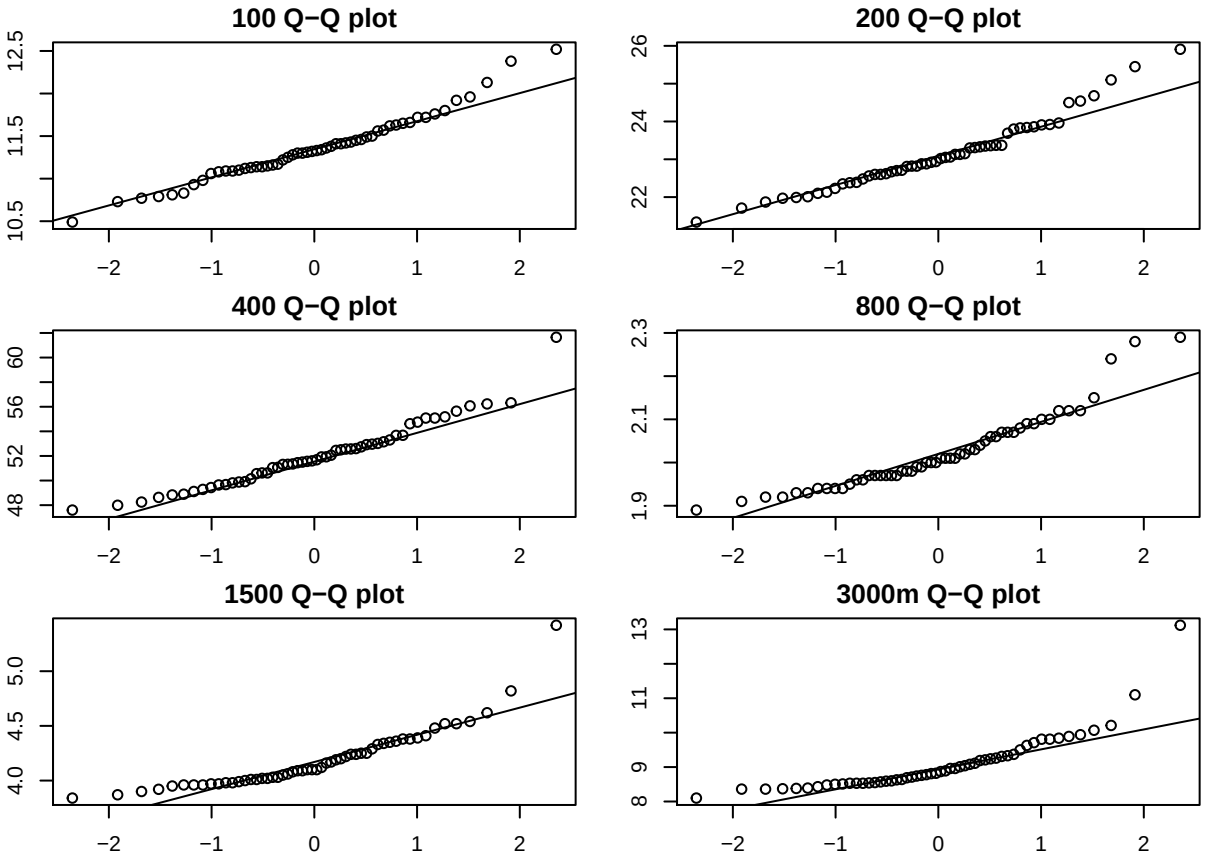
```r
library(ggplot2)
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

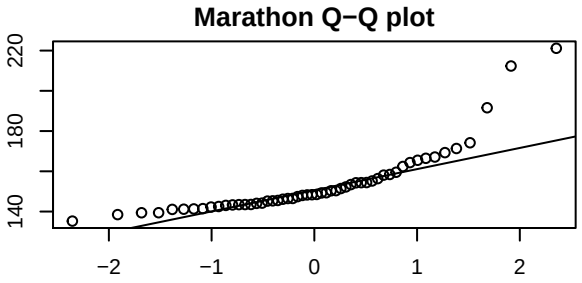
```r
# Scatter plot matrix.
ggpairs(data[,2:8])
```



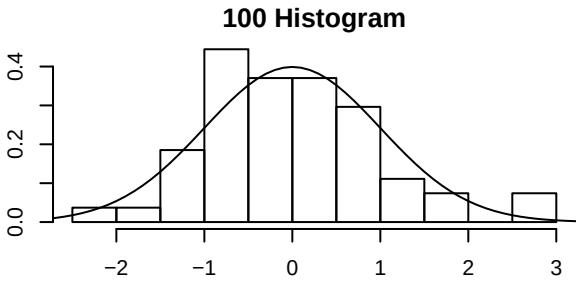

```r
# Q-Q plots.
par(mfrow=c(3,2))

for (i in 2:8)
{
  par(mar = c(2, 2, 2, 2))
  qqnorm(data[, i], main=paste(colnames(data)[i], "Q-Q plot"))
  qqline(data[, i])
}
```

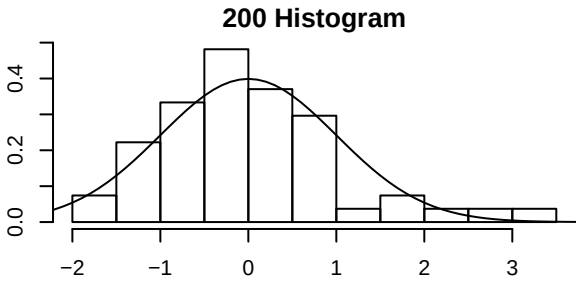


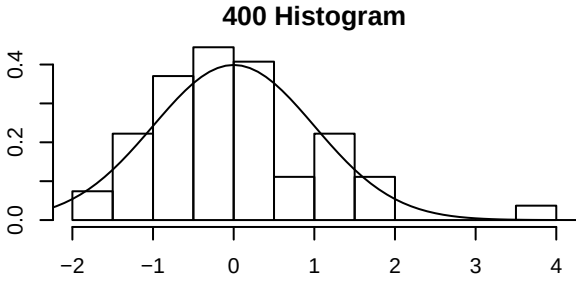

```
# Histograms with normal distribution.
par(mfrow=c(3,2))
```

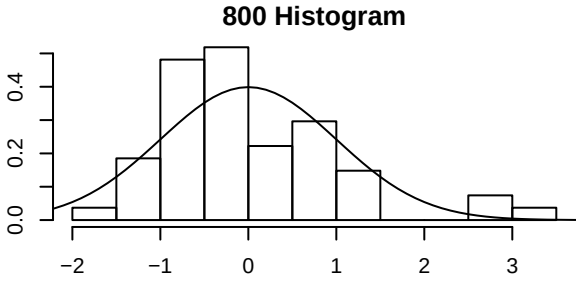


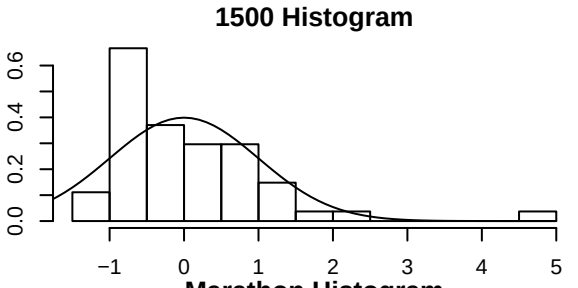

```
# Values for the normal distribution.
x = seq(-5, 5, 0.1)
y = dnorm(x)

for (i in 2:8)
{
  par(mar = c(2, 2, 2, 2))
  hist(scale(data[, i]),
       freq=FALSE,
       breaks=10,
       main=paste(colnames(data)[i], "Histogram"))
  lines(x, y)
}
```

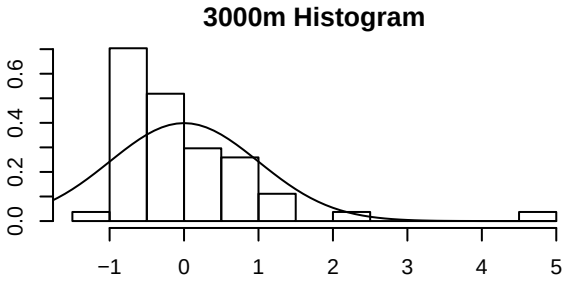


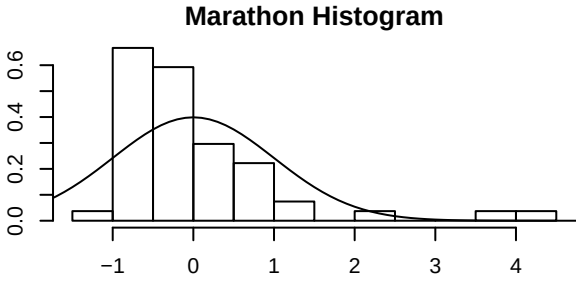












Question 2: Relationships between the variables

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

A structure over the variable is visible on the correlation heatmap below. It's clear that there are two groups of tracks. The ones that are about 400 meters and below and the ones that are above. These two groups are the ones that correlates the highest with each other. It's also worth noting that the minimum correlation between the tracks is 0.66, which means that all the tracks are related to each other in some manner even if they seem to have different groups.

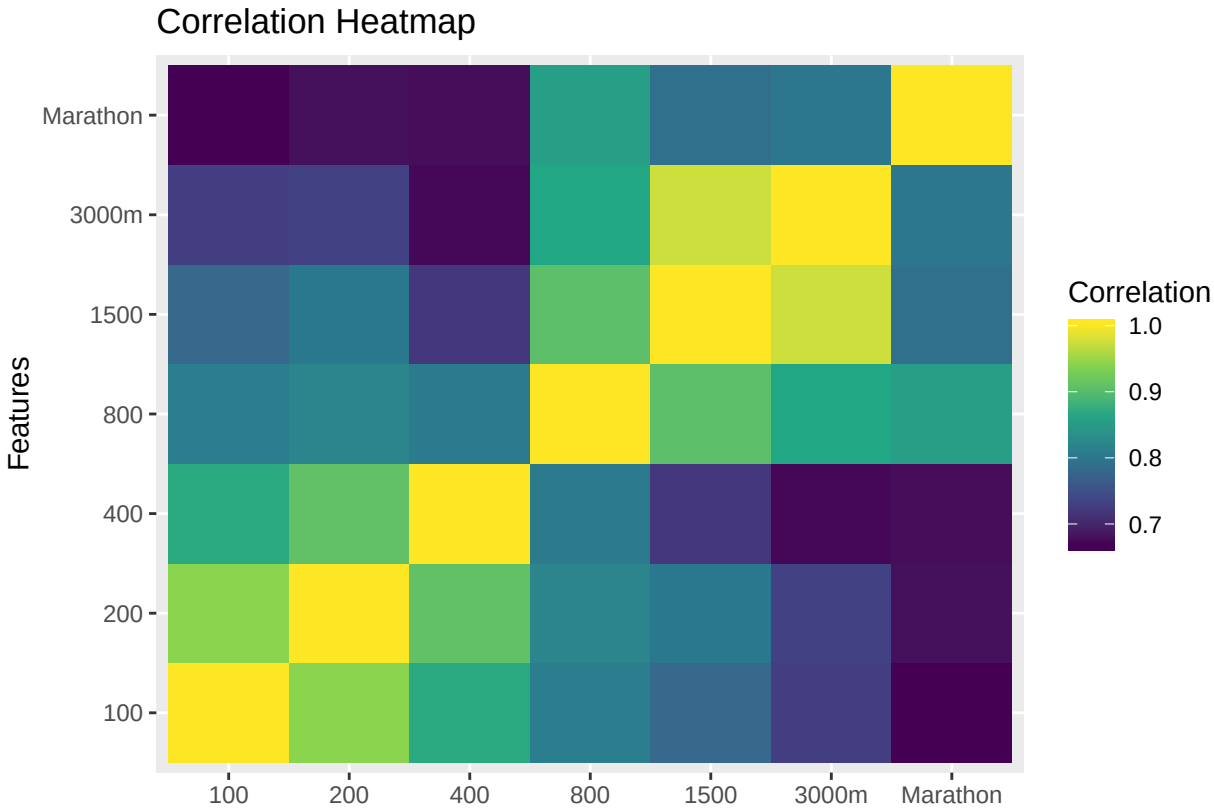
```
library(reshape2)

corr_matrix = cor(data[, 2:8])
cov_matrix = cov(data[, 2:8])

print(corr_matrix)
```

```
##               100       200       400       800      1500     3000m
## 100      1.0000000 0.9410886 0.8707802 0.8091758 0.7815510 0.7278784
## 200      0.9410886 1.0000000 0.9088096 0.8198258 0.8013282 0.7318546
## 400      0.8707802 0.9088096 1.0000000 0.8057904 0.7197996 0.6737991
## 800      0.8091758 0.8198258 0.8057904 1.0000000 0.9050509 0.8665732
## 1500     0.7815510 0.8013282 0.7197996 0.9050509 1.0000000 0.9733801
## 3000m    0.7278784 0.7318546 0.6737991 0.8665732 0.9733801 1.0000000
## Marathon 0.6689597 0.6799537 0.6769384 0.8539900 0.7905565 0.7987302
##           Marathon
## 100      0.6689597
## 200      0.6799537
## 400      0.6769384
## 800      0.8539900
## 1500     0.7905565
## 3000m    0.7987302
## Marathon 1.0000000
```

```
print(cov_matrix)
```

```
##                100        200        400         800       1500
## 100      0.15531572  0.3445608  0.8912960 0.027703564 0.08389119
## 200      0.34456080  0.8630883  2.1928363 0.066165898 0.20276331
## 400      0.89129602  2.1928363  6.7454576 0.181807932 0.50917683
## 800      0.02770356  0.0661659  0.1818079 0.007546925 0.02141457
## 1500     0.08389119  0.2027633  0.5091768 0.021414570 0.07418270
## 3000m    0.23388281  0.5543502  1.4268158 0.061379315 0.21615514
## Marathon 4.33417757 10.3849876 28.9037314 1.219654647 3.53983732
##                3000m   Marathon
## 100       0.23388281   4.334178
## 200       0.55435017  10.384988
## 400       1.42681579  28.903731
## 800       0.06137932   1.219655
## 1500      0.21615514   3.539837
## 3000m     0.66475793  10.706091
## Marathon 10.70609113 270.270150
```

```
melted_corr = melt(corr_matrix)
melted_cov = melt(cov_matrix)

p = ggplot(data=melted_corr, aes(x=Var1, y=Var2, fill=value)) +
    geom_tile() +
    scale_fill_viridis_c() +
    labs(x="", y="Features", fill="Correlation", title="Correlation Heatmap")
print(p)
```



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

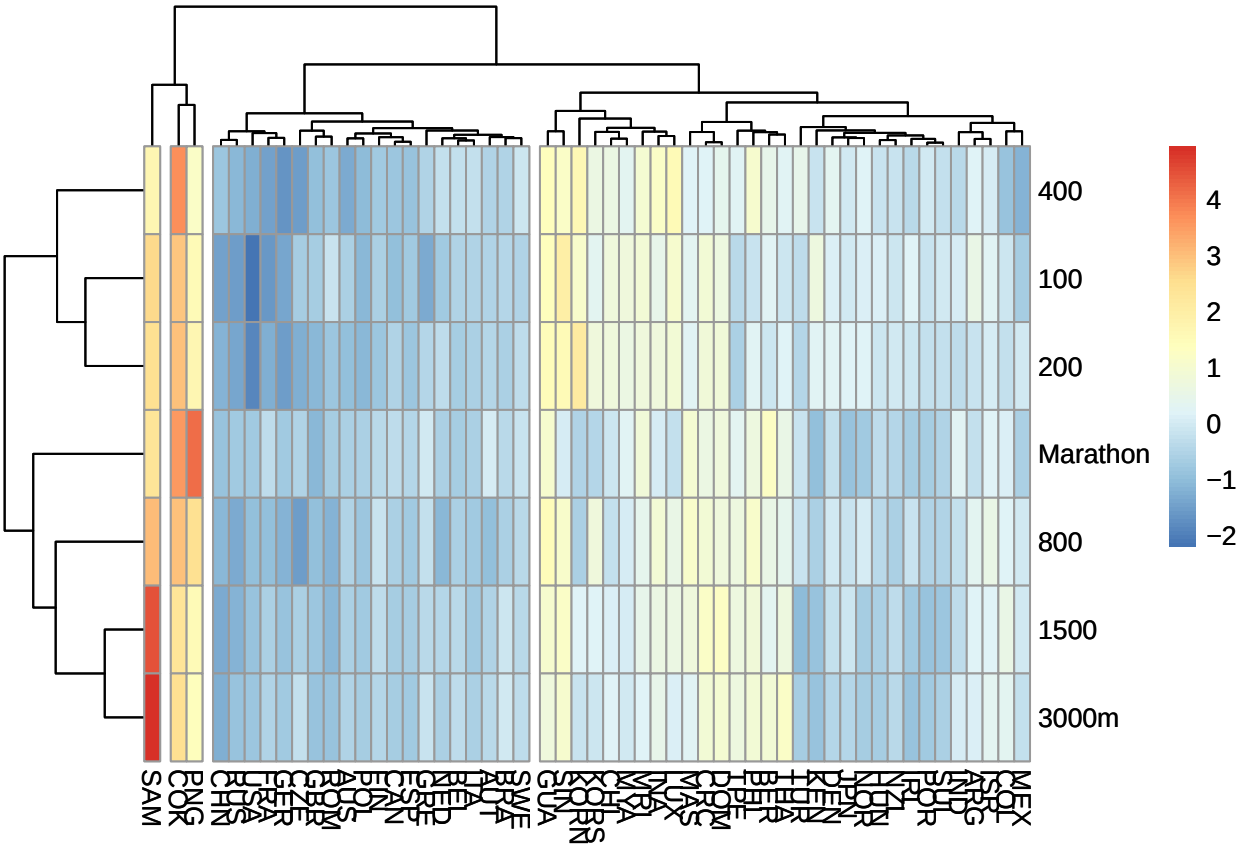
The scatter plot matrix was shown above on the first question. The extreme values are present always on the upper quantiles of each variables. Meaning that there are some countries that excels on each category.

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.

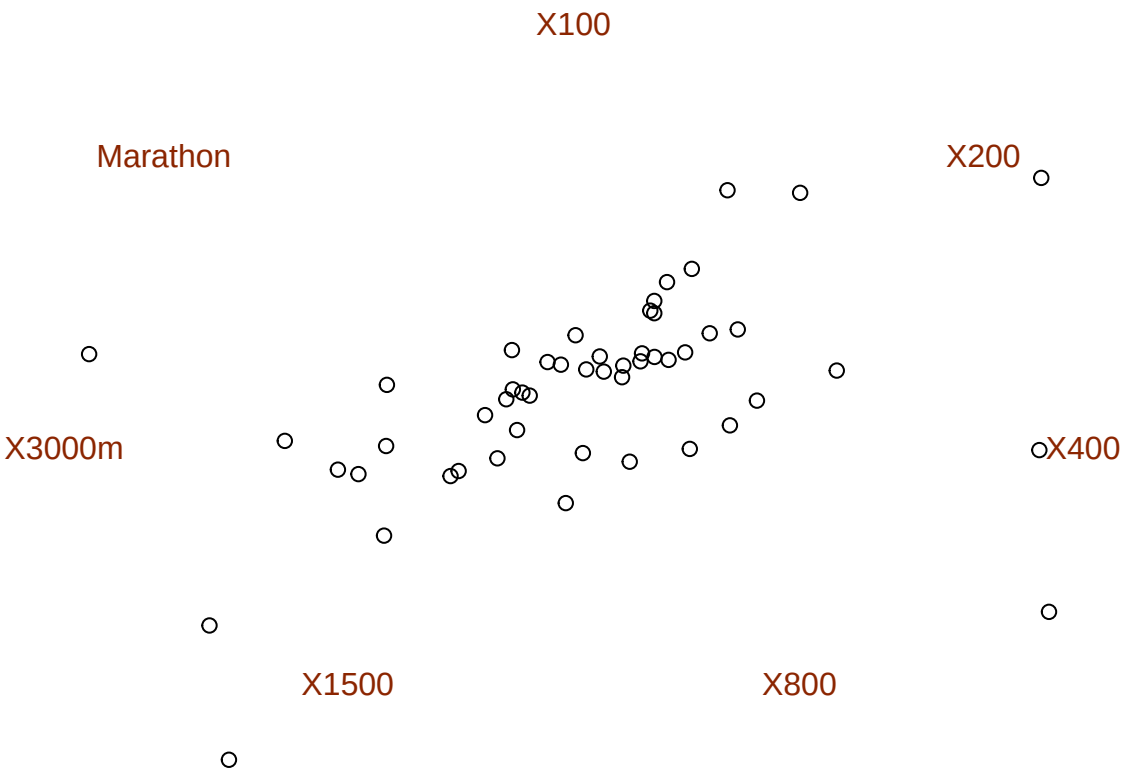
From the hierarchical clustering plot and heatmap plot we can see that there are three main clusters. The first one corresponds to some three countries which seems to be outliers (SAM, COK, PNG). The next cluster seems to be conformed by developed nations like Sweden or the USA and the final cluster is conposed of developing nations like Mexico, Colombia and Portugal.

The second visualization is based on dimensional anchors. It projects a high dimensional space into a 2d circle where each anchor (feature) is evenly spaced on a unit circle, then observations are plotted with respect to their values. From this plot we can see that the most variability comes from the two "groups" mentioned before. Tracks from 400 meters and less and those with higher meters. This might suggest that doing some dimensionality reduction over the dataset is not a bad idea.

```
library(magrittr)
library(pheatmap)
library(Radviz)


data2 = data
rownames(data2) = data2$Country
data2$Country = NULL

p = data2 %>% scale() %>% t() %>% pheatmap(cutree_cols=4)
print(p)
```




```
colnames(data2) = c("X100", "X200", "X400", "X800", "X1500", "X3000m", "Marathon")
das = colnames(data2)
S = make.S(das)
rv = do.radviz(scale(data2), S)
plot(rv, point.shape=1)
```



Question 3: Examining for extreme values

a) Look at the plots (esp. scatterlots) generated in the previous question. Which 3-4 countries appear most extreme? Why do you consider them extreme?

From the scatter plots and the Q-Q plots there are four countries that stand out the most on the 800 meters track. The countries are SAM, COK, PNG and GUA. These countries are the same ones thata are the most slow on the feature heatmap and hierarchical cluster presented above. They are considered "extreme" because they are the slowest countries overall.

```
idx = sort(data$`800`, decreasing=TRUE, index.return=TRUE)$ix
countries = data$Country[idx[1:4]]
print(countries)
```

```
## [1] SAM COK PNG GUA
## 54 Levels: ARG AUS AUT BEL BER BRA CAN CHI CHN COK COL CRC CZE DEN ... USA
```

One approach to measuring "extremisim" is to look at the distance (needs to be defined!) between an observation and the sample mean vector, i.e. we look how far one is from the average. Such a distance can be called an *multivariate residual* for the given observation.

b) The most common residual is the Euclidean distance between the observation and sample mean vector, i.e.

$$d(\vec{x}, \bar{x}) = \sqrt{(\vec{x} - \bar{x})^T (\vec{x} - \bar{x})}$$

The distance can be immediately generalized to the $L^r, r > 0$ distance as

$$d_{L^r}(\vec{x}, \bar{x}) = \left(\sum_{i=1}^{p} |\vec{x}_i - \bar{x}_i|^r \right)^{1/r}$$

where p is the dimension of the osbervation (here $p = 7$).

Compute the squared Eculidean distance (i.e. $r = 2$) of the observation from the sample mean for all 55 countries using `R` matrix operation. First center the raw data by the means to get $\vec{x} - \bar{x}$ for each country. then do a calculation with matrices that will result in a matrix that has on its diagonal the requested squared distance for each country. Copy this diagonal to a vector and report on the five most extreme countries. In this question you **MAY NOT** use any loops.

```
euclidean_distance = function(X)
{
  mu = matrix(1, nrow=1, ncol=dim(X)[1]) %*% X / dim(X)[1]
  mu_broadcast = matrix(mu, nrow=dim(X)[1], ncol=dim(X)[2], byrow=TRUE)
  X_centered = X - mu_broadcast
  X_distance = sqrt(diag(X_centered %*% t(X_centered)))

  return(X_distance)
}

distances_ed = euclidean_distance(as.matrix(data[, 2:8]))
idxs = sort(distances_ed, decreasing=TRUE, index.return=TRUE)$ix
countries = data$Country[idxs[1:5]]
print(countries)
```

```
## [1] PNG COK SAM BER GBR
## 54 Levels: ARG AUS AUT BEL BER BRA CAN CHI CHN COK COL CRC CZE DEN ... USA
```

c) The different variables have different scales so it is possible that the distances can be dominated by some few variables. To avoid this we can use the squared distance

$$d^2_{\mathbf{V}}(\vec{x}, \bar{x}) = (\vec{x} - \bar{x})^T \mathbf{V}^{-1} (\vec{x} - \bar{x})$$

where $\mathbf{V}$ is a diagonal matrix with variances of the appropiate variables on the diagonal. The effect, is that for each variable the squared distance is divided by its variance and we have a scaled independent distance.

It is simple to compute this measure by standardizing the raw data with both means (centering) and standard deviations (scaling), and then compute the euclidean distance for the normalized data. Carry out these computations and conclude which countries are the most extreme ones. How do your conclusions compare with the unnomalized ones.

The countries remain the same except for BER and GBR which are replaced by USE and SIN. This is a more "fair" since the countries are being compared on the same scale.

```
euclidean_distance_centered = function(X, varcov)
{
  mu = matrix(1, nrow=1, ncol=dim(X)[1]) %*% X / dim(X)[1]
  mu_broadcast = matrix(mu, nrow=dim(X)[1], ncol=dim(X)[2], byrow=TRUE)
  X_centered = X - mu_broadcast
  V = diag(diag(varcov))
  X_distance = sqrt(diag(X_centered %*% solve(V) %*%t(X_centered)))

  return(X_distance)
}
```

```
distances_edc = euclidean_distance_centered(as.matrix(data[, 2:8]), cov_matrix)
idxs = sort(distances_edc, decreasing=TRUE, index.return=TRUE)$ix
countries = data$Country[idxs[1:5]]
print(countries)
```

```
## [1] SAM COK PNG USA SIN
## 54 Levels: ARG AUS AUT BEL BER BRA CAN CHI CHN COK COL CRC CZE DEN ... USA
```

d) The most common statistical distance is the *Mahalanobis distance*

$$d_{\mathrm{M}}^2(\vec{x}, \bar{x}) = (\vec{x} - \bar{x})^T \mathbf{C}^{-1} (\vec{x} - \bar{x})$$

where $\mathbf{C}$ is the sample convariance matrix calculated from the data. With this measure we also use the relationships (covariances) between the variables (and not only the marginal variances as $d_{\mathbf{V}}$ does). Compute the Mahalanobis distance, which countries are most extreme now?

The same behaviour happens again, the countries SAM, COK and PNG stay on the most extreme and new two countries join the set. These countries are KORN and MEX.

```
mahalanobis_distance = function(X, varcov)
{
  mu = matrix(1, nrow=1, ncol=dim(X)[1]) %*% X / dim(X)[1]
  mu_broadcast = matrix(mu, nrow=dim(X)[1], ncol=dim(X)[2], byrow=TRUE)
  X_centered = X - mu_broadcast
  X_distance = sqrt(diag(X_centered %*% solve(varcov) %*%t(X_centered)))

  return(X_distance)
}

distances_md = mahalanobis_distance(as.matrix(data[, 2:8]), cov_matrix)
idxs = sort(distances_md, decreasing=TRUE, index.return=TRUE)$ix
countries = data$Country[idxs[1:5]]
print(countries)
```

```
## [1] SAM  PNG  KORN COK  MEX
## 54 Levels: ARG AUS AUT BEL BER BRA CAN CHI CHN COK COL CRC CZE DEN ... USA
```

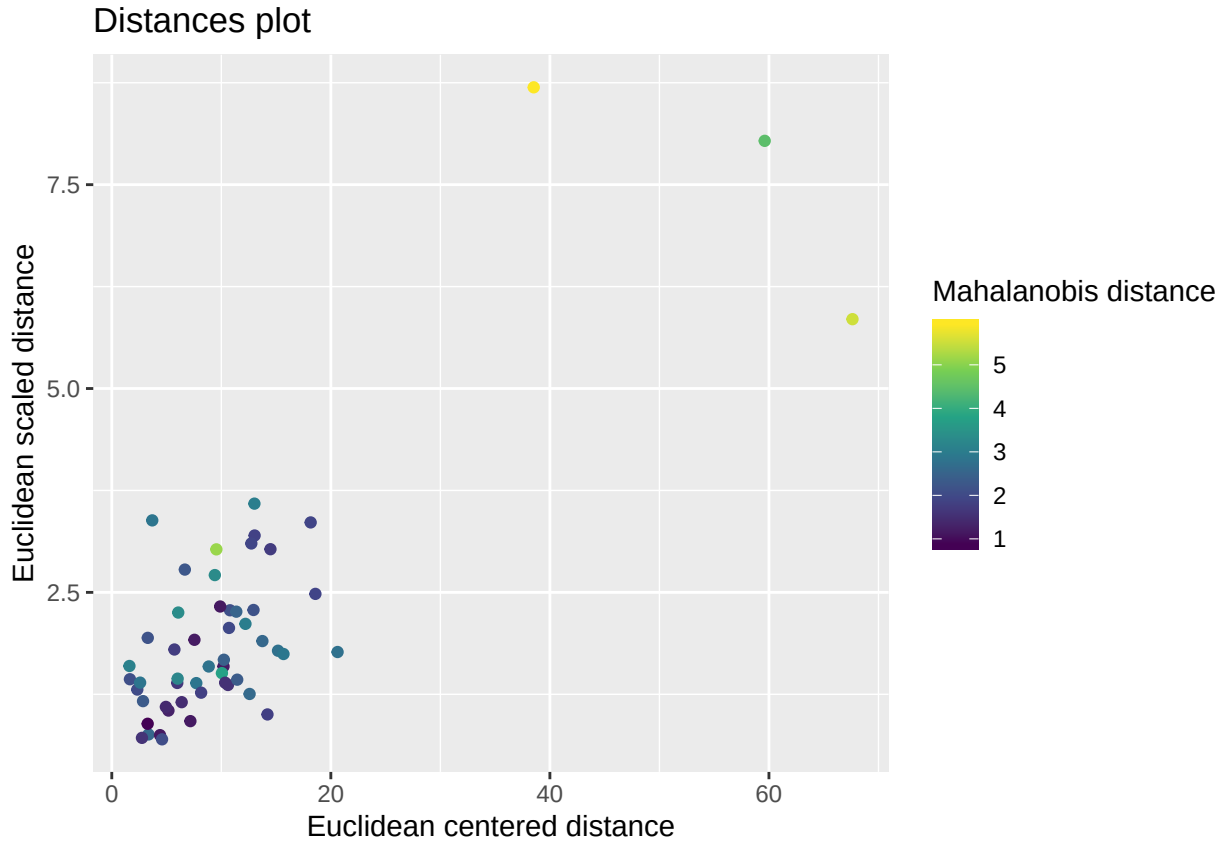
e) Compare the results in b)-d). Some of the countries are the upper end with all the measures and perhaps they can be classified as extreme. Discuss this. But also notice the different mesaures give rather different results (how does Sweden behave?). Summarize this graphically. Produce Czekanowski's diagram using e.g. `RMaCzek` package. In case of problems please describe them.

The countries that always end up together are SAM, COK and PNG. It can see below from the two scatter plots that they are further away from the rest of the countries. The other countries stays close together in a cluster on the bottom left of the scatter plot.

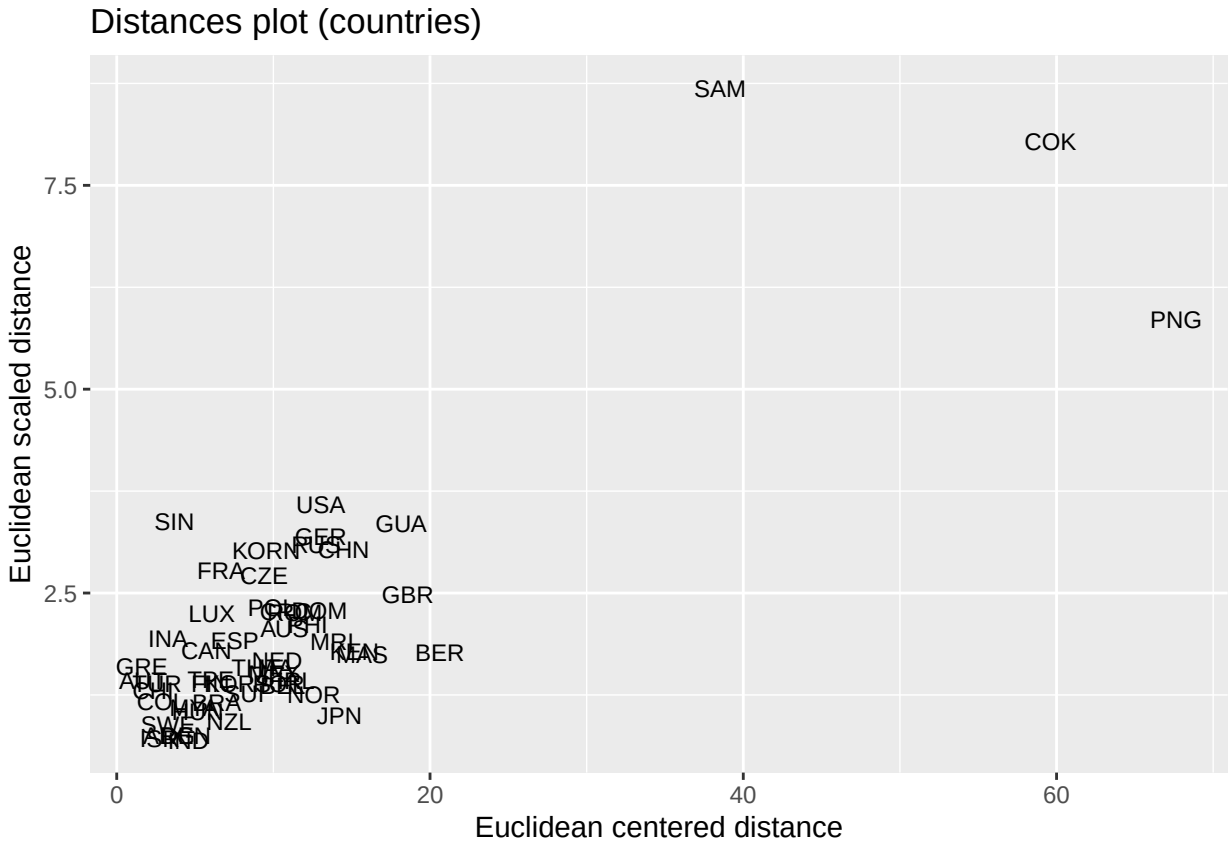
In the case of Sweden, it has a "high" euclidean distance but the other two metrics are small. In general, Sweden did pretty well on all metrics. Is one of the countries with the smallest times on all tracks regarding each of the three distances.

```
distance_data = list(Country=data$Country,
                     euclid_center=distances_ed,
                     euclid_scale=distances_edc,
                     mahalanobis=distances_md)
distance_data = data.frame(distance_data)
```

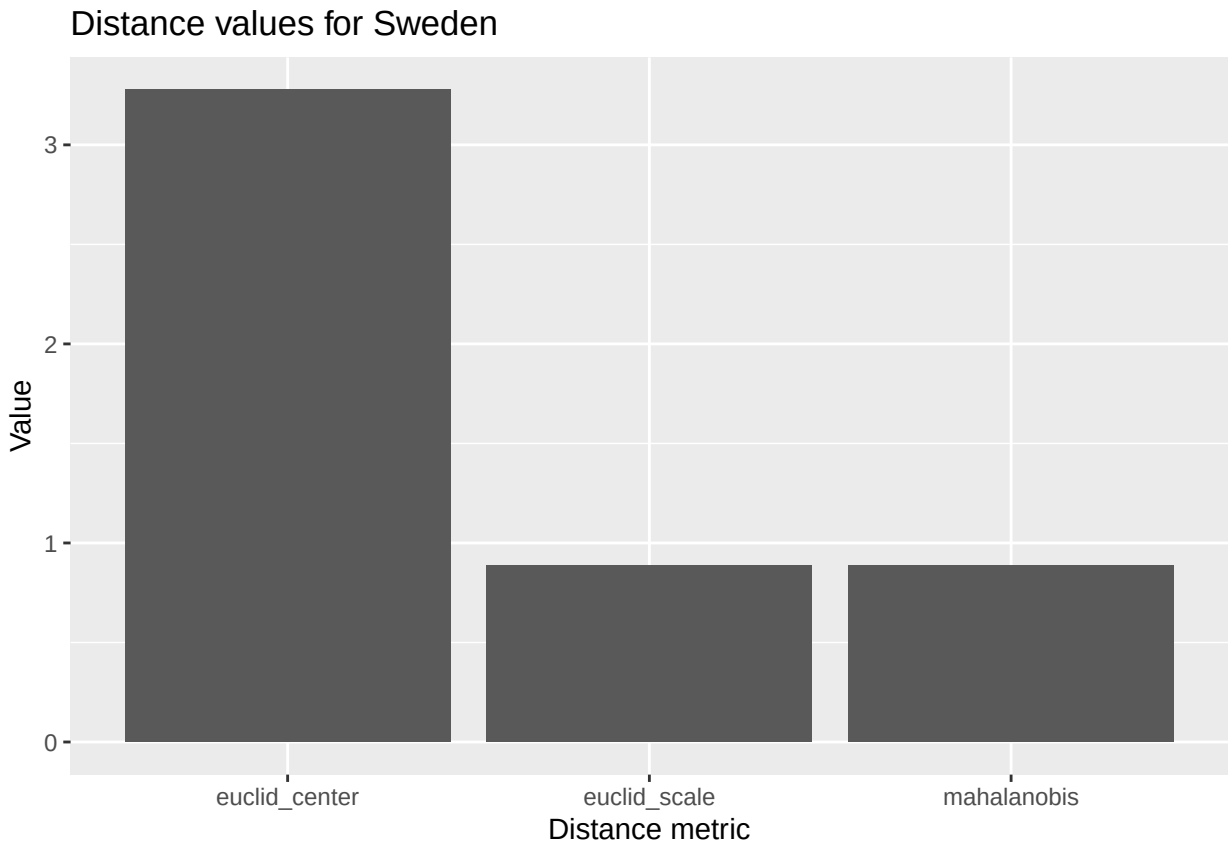
```
p = ggplot() +
    geom_point(aes(x=distance_data$euclid_center,
                   y=distance_data$euclid_scale,
                   colour=distance_data$mahalanobis)) +
    scale_color_viridis_c() +
    labs(x="Euclidean centered distance",
         y="Euclidean scaled distance",
         colour="Mahalanobis distance",
         title="Distances plot")

print(p)
```




```
p = ggplot() +
    geom_text(aes(x=distance_data$euclid_center,
                  y=distance_data$euclid_scale,
                  label=distance_data$Country),
              size=3) +
    scale_color_viridis_c() +
    labs(x="Euclidean centered distance",
         y="Euclidean scaled distance",
         colour="Mahalanobis distance",
         title="Distances plot (countries)")

print(p)
```




```
swe_mask = distance_data$Country == "SWE"
x = colnames(distance_data)[2:4]
y = as.numeric(distance_data[swe_mask, 2:4])

p = ggplot() +
    geom_col(aes(x=x,
                 y=y)) +
    labs(x="Distance metric", y="Value", title="Distance values for Sweden")

print(p)
```




```
library(RMaCzek)
```

```
## Registered S3 method overwritten by 'seriation':
##   method         from
##   reorder.hclust gclus
```

```
czek_matrix_res = czek_matrix(data[, 2:8])
plot(czek_matrix_res)
```

Czekanowski's diagram