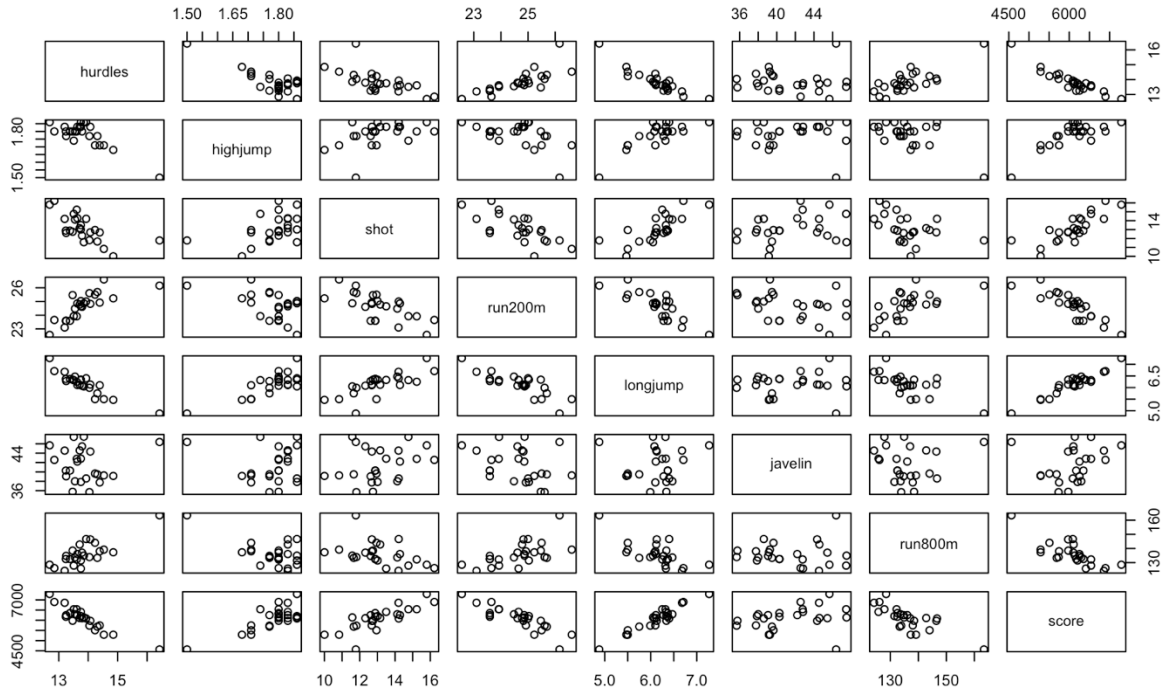
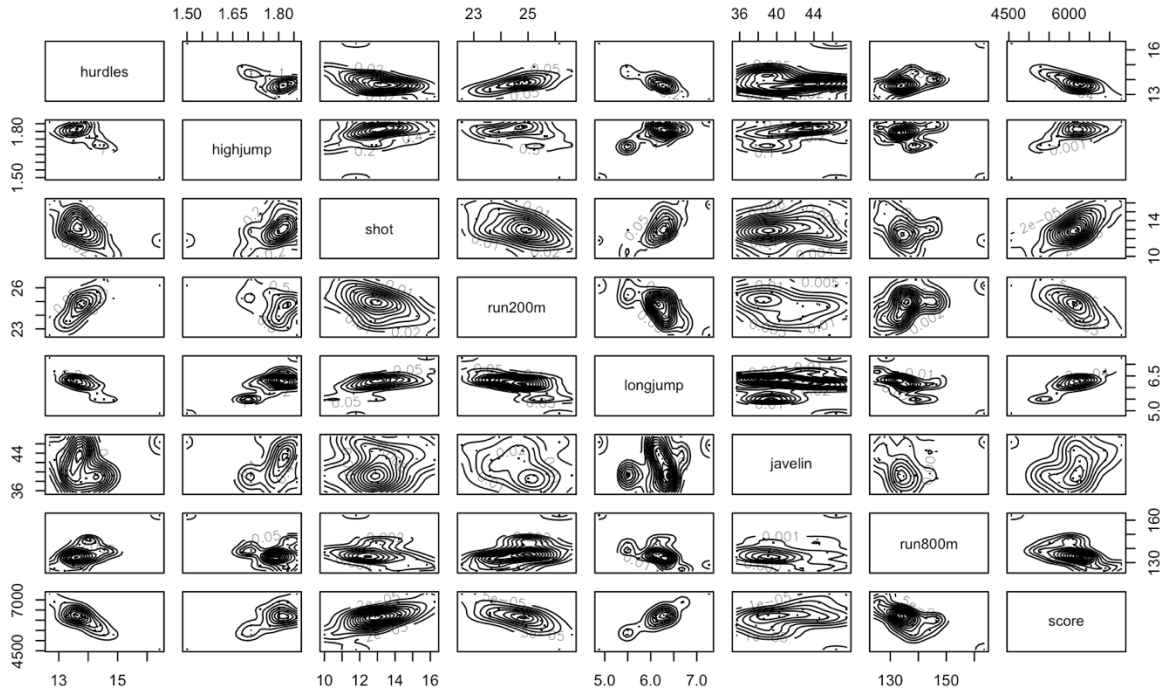
W4415 Multavariate Statistical Inference Homework3

Q1

```
par(mfrow=c(1,3))
pairs(heptathlon)
pairs(heptathlon,panel=function(x,y,...){
  points(x,y,...)
  contour(kde2d(x,y),add = TRUE)
},pch=".",cex=1.5)
```

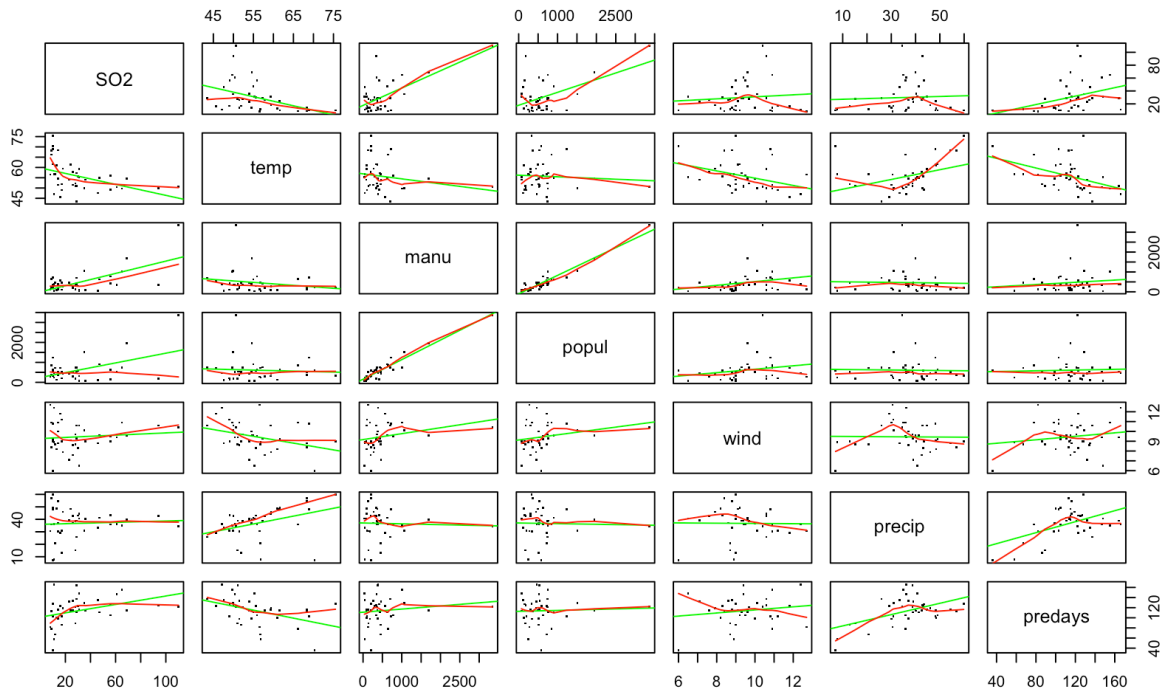




The estimated bivariate density function plots are more useful than unenhanced scatterplots because they show how data is centered. Although scatterplot matrices are good at highlighting outliers in a multivariate data set. But when to identify regions in the plot where there are high or low densities of observations that may indicate the presence of distinct groups of observations; i.e., “clusters”.

Q2

```
data("USairpollution")
wea<-colnames(USairpollution)
pairs(USairpollution,panel=function(x,y,...){
  points(x,y,...)
  abline(lm(y~x),col="green")
  lines(lowess(y~x),col="red")
},pch=".",cex=1.5)
```



Green lines indicate fitted linear regression and red lines indicate fitted locally weighted regression.
(1) The graphs above are useful to decide on the most appropriate model for determining the variables most predictive of SO2 levels. Fitted linear regression lines suggest a linear relationship between SO2 and variable "manu" and "popul", while locally weighted regression may be over-fitted.
(2) However, when determining the relationship between SO2 and "precip", both lines work poorly as there may exists high degree of relationship between these 2 variables.
In all, the information provided by the graphs is not enough to determine the most appropriate model and hence in-depth analysis is needed.

```
cor(USairpollution)[,1]
```

```
SO2         temp         manu        popul         wind       precip      predays
1     -0.43360020   0.64476873   0.49377958   0.09469045   0.05429434   0.36956363
```

After calculating the correlation between SO2 and other variables, there exists a high correlation between SO2 and "manu" and "popul", which backs up my assumption.

Q3

(1)

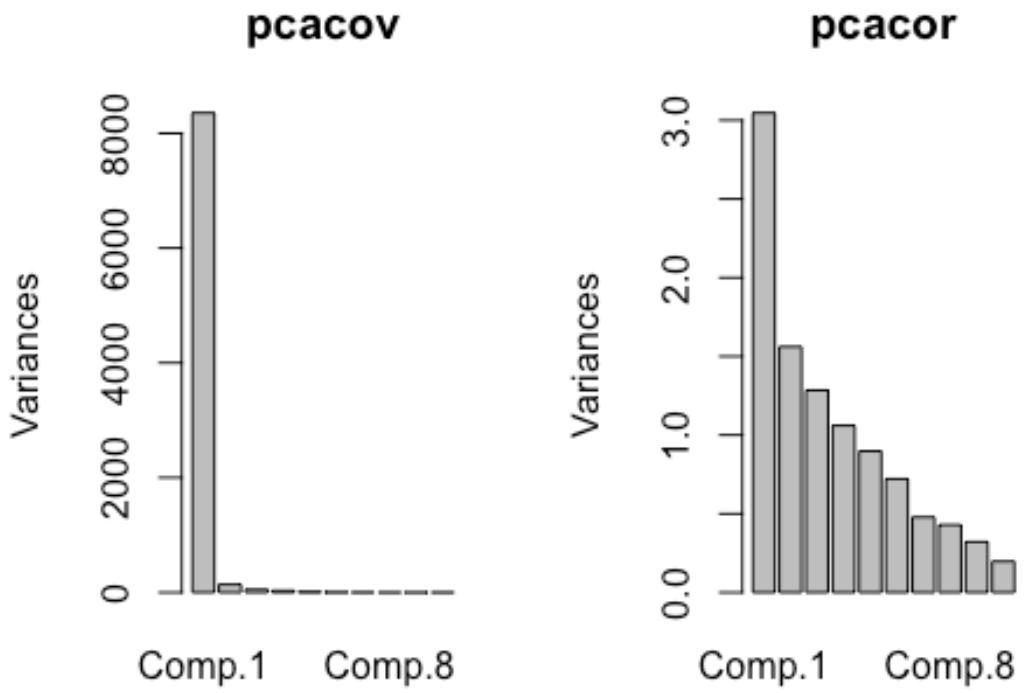
```
ze <- read.delim("zehnkampf.dat")
zep<-ze[,-11]
zepcor<-cor(zep)
zepcov<-cov(zep)

pcacov<-princomp(covmat=zepcov)
pcacor<-princomp(covmat=zepcor)

par(mfrow=c(1,2))
```

```
screeplot(pcacov)
screeplot(pcacor)
```

It is more advisable to use correlation matrix to do PCA. Examining the results, we see that each of the principal components of the covariance matrix is largely dominated by a single variable, whereas those for the correlation matrix have moderate-sized coefficients on several of the variables. And the first component from the covariance matrix accounts for almost 99% of the total variance of the observed variables. The components of the covariance matrix are completely dominated by the fact that the variance of the pole vault (stab) variable is over 60 times larger than the variance of any of the 9 other variables. Consequently, the principal components from the covariance matrix simply reflect the order of the sizes of the variances of the observed variables.



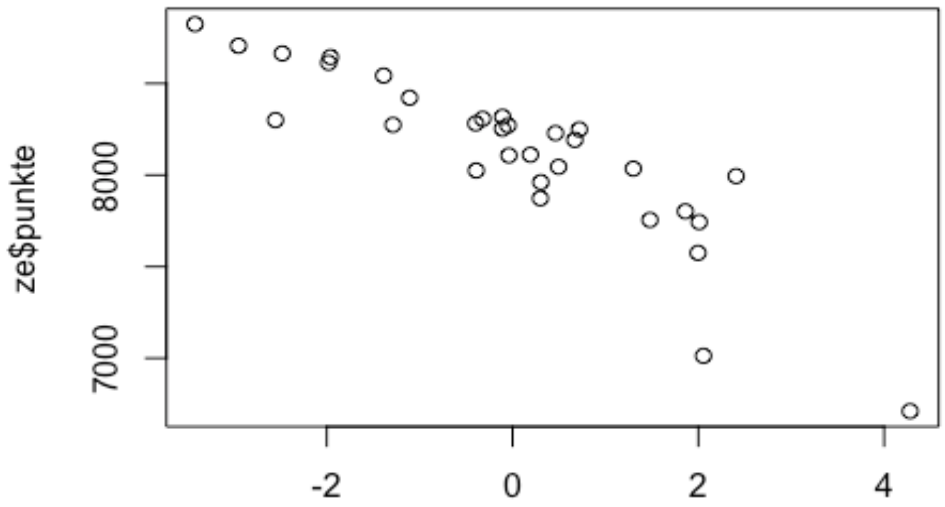
(2)

a)

```
pcacor$loadings[,1]
```

```
##        m100        weit       kugel        hoch        m400        hurd
##  0.46997571 -0.37650680 -0.21632796 -0.06622491  0.31620762  0.41304416
##        disc        stab       speer       m1500
## -0.33240870 -0.28764061 -0.32123489 -0.13843656
```

```
plot(scale(zep,center=TRUE,scale=TRUE)%*%pcacor$loadings[,1],ze$punkte)
```



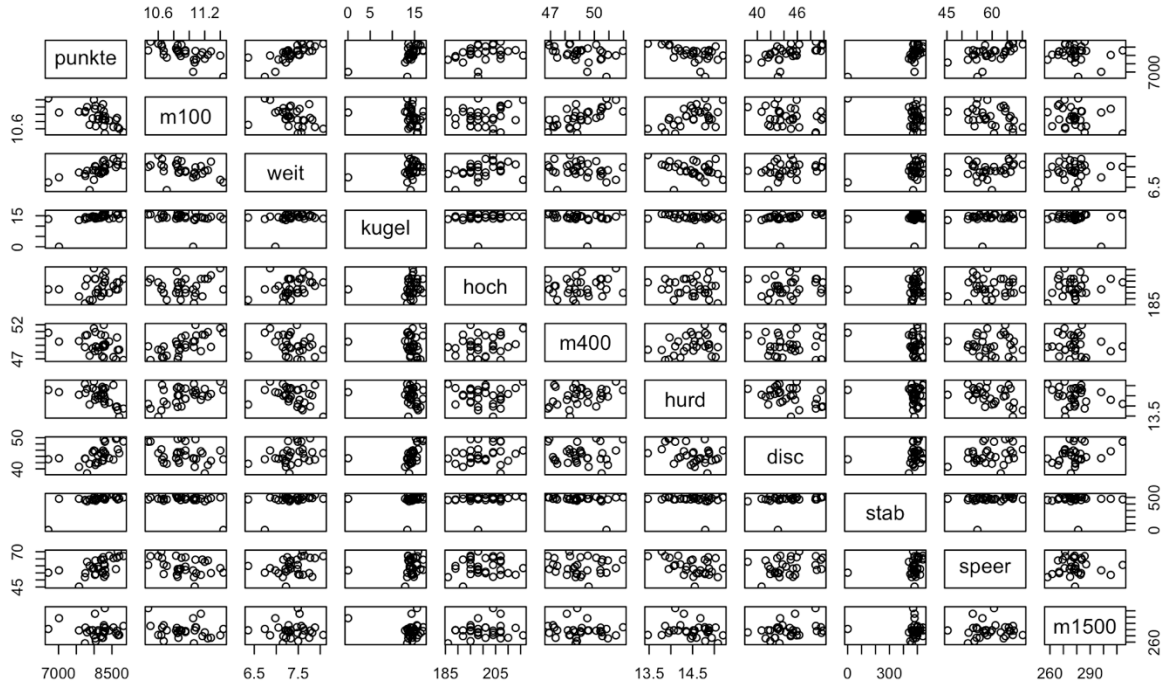
The scatterplot above indicated that the first principal component is in good agreement with the score assigned to the athletes. From the result of PCA's loadings, the 1st component places a higher weight on 100m-sprint (m100), so athletes should behave well in this discipline in order to get a higher score.

Remarks:

(1) As scores are not provided when PCA use correlation or covariance matrix directly, I have to use the loadings to analyze instead. As loadings=eigenvectors x $\sqrt{eigenvalue}$, with eigenvectors being directions and eigenvalues being variance along these directions, the weight of each original discipline matters. That's why I use loadings to interpret the results;

(2) As 400m-sprint and hurdles both have a high weight in the 1st component, I think the athlete should perform well in these disciplines too.

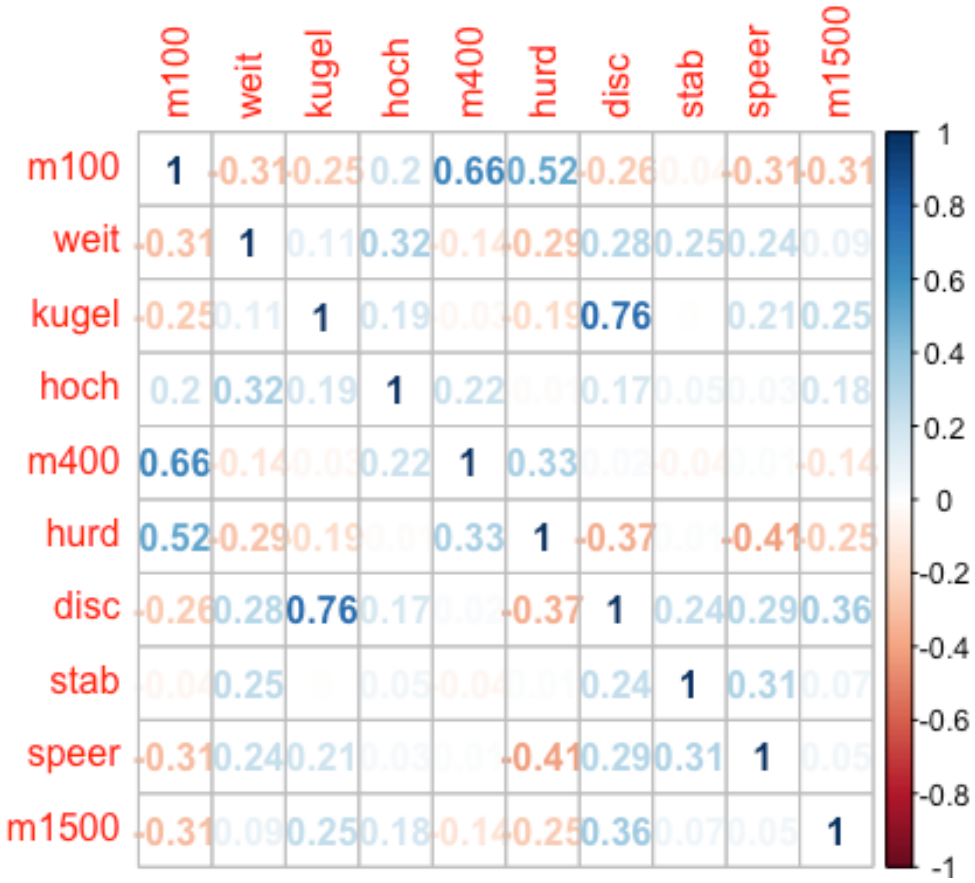
b)



The visualization shows that Shotput (kugel) and Pole vault (stab) are poorly displayed, because AFANASYEV did poor (got 0) in pole vault and POSERINA did poor (got 0) in Shotput while the others perform way better than them.

c)

```
library(corrplot)
par(mfrow=c(1,1))
corrplot(cor(ze[-c(30:31),-11]),method="number")
```



After excluding the extreme cases (AFANASYEV and POSERINA),
(1) Shotput (kugel) and Discus (disc) show a high correlation between each other. Maybe because these disciplines all require power;
(2) 100m-sprint (m100), 400m-sprint (m400) and 110m-hurdles (hurd) also display a high correlation, as they all require speed and explosive force.

d)

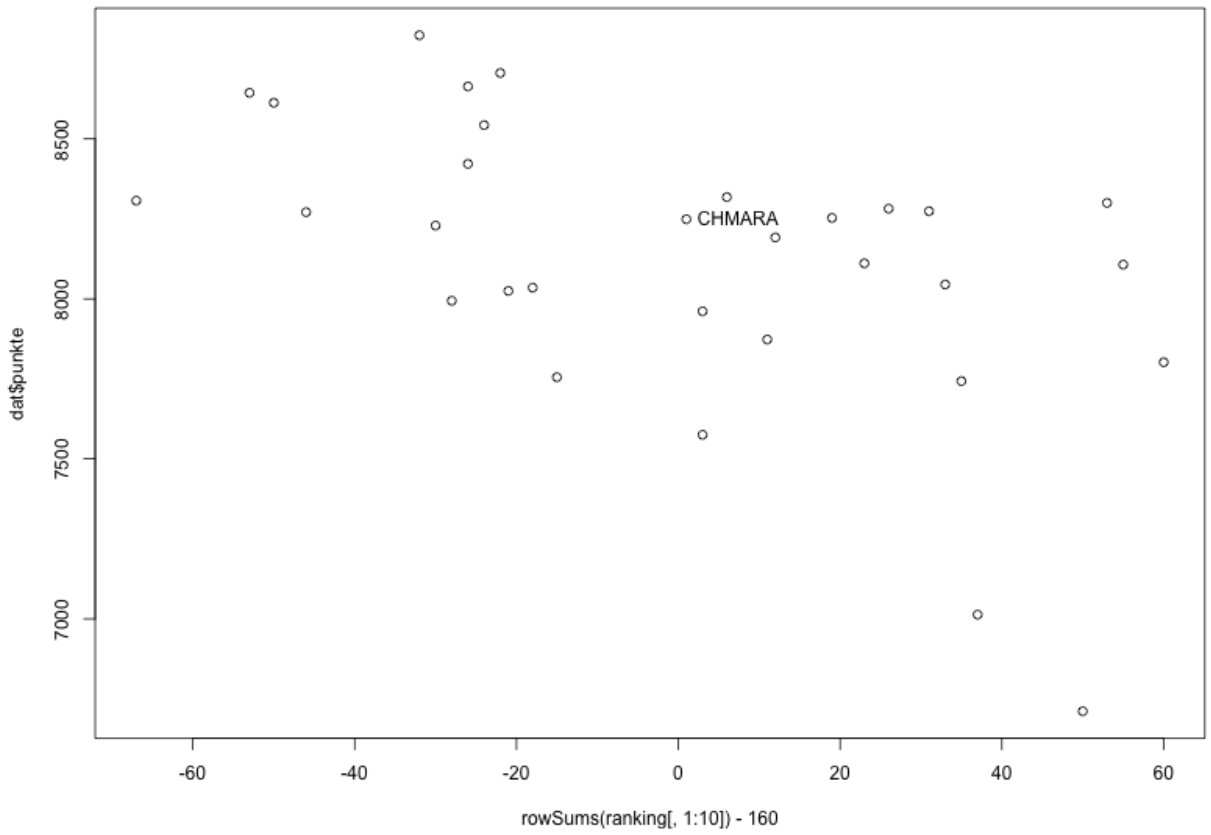
Javelin (speer), Discus (disc) and 110m-hurdles (hurd) show a negative correlation, suggesting that athletes cannot do well on the former two and the latter. Because Javelin and Discus require more on skills and the strength of upper body, while 100m-hurdles focus more on lower body power.

e)

Pole vault (stab) has a low correlation with most of other disciplines, suggesting that it is hard to predict the athlete's performance based on his performance on Pole vault.

(3)

```
rankcol<-vector()
ranking<-vector()
for (i in 1:11){
  rankcol<-order(ze[,i],decreasing = TRUE)
  ranking<-cbind(ranking,rankcol)
  rankcol<-vector()
}
plot(rowSums(ranking[,1:10])-160,ze$punkte)
identify(rowSums(ranking[,1:10])-160,ze$punkte,labels=rownames(ze),plot=TRUE)
```

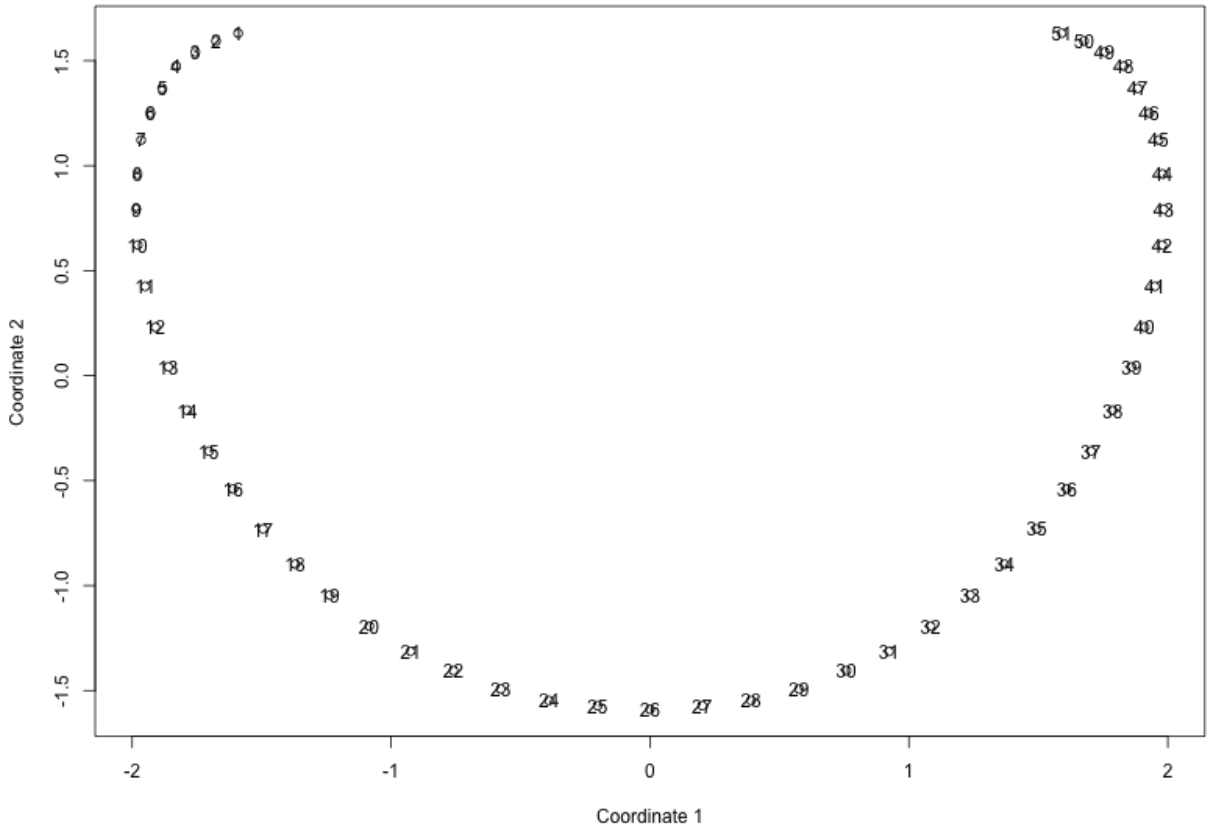


If place the same weight on each of the discipline, then based on the ranking of each discipline, CHMARA is the one who is the average athlete whose average ranking is in the middle of all.

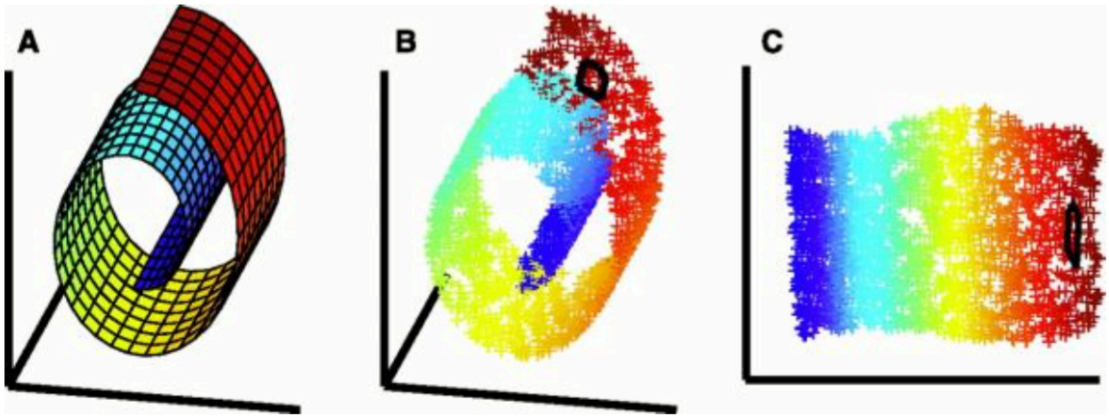
Q4

```
S<-matrix(nrow=51,ncol=51)
for(i in 1:51){
  for (j in 1:51){
    if (row(S)==col(S)) S[i,i]=9
    if (i-j>=1 && i-j<=3) S[i,j]=8
    if (i-j>=4 && i-j<=6) S[i,j]=7
```

```
    if (i-j>=7 && i-j<=9) S[i,j]=6
    if (i-j>=10 && i-j<=12) S[i,j]=5
    if (i-j>=13 && i-j<=15) S[i,j]=4
    if (i-j>=16 && i-j<=18) S[i,j]=3
    if (i-j>=19 && i-j<=21) S[i,j]=2
    if (i-j>=22 && i-j<=24) S[i,j]=1
    if(i-j>=25) S[i,j]=0
  }
}
sigma<-matrix(nrow=51,ncol=51)
for (i in 1:51){
  for (j in 1:51){
    sigma[i,j]=sqrt(S[i,i]+S[j,j]-2*S[i,j])
  }
}
for (i in 1:51){
  for (j in 1:51){
    sigma[i,j]=sigma[j,i]
  }
}
sigma_mds<-cmdscale(sigma,k=2,eig=TRUE)
plot(sigma_mds$points[,1],sigma_mds$points[,2],xlab="Coordinate
1",ylab="Coordinate 2")
text(sigma_mds$points[,1],sigma_mds$points[,2],labels=colnames(S))
```



(1) The plot above shows that, the dissimilarity between each Object i and j increases as |i-j| increases. Because positions of Objects show a patter of symmetry, for example, Object 1 and 51 are both on the ends of the line, they show a symmetric patter on Coordinate 1 as well. Therefore, they have the same value on Coordinate 2.



pictures taken from Lecture Note WK5A MDS I

(2) Also, when considering the nature of MDS, it actually describes points on a manifold with high-dimensional data given. Although the given points lie on a straight line, when MDS treats it as a curve distance just like picture B above, it will display a curve on as 2-D plot.
(3) In this question, the similarity matrix is not calculating points' Euclidean distance directly but actually placing arbitrary weights on them, as the input for MDS is proximities, the plot given can be different from a straight line.