Multivariate Statistical Methods - Lab 03

Maximilian Pfundstein (maxpf364), Hector Plata (hecpl268), Aashana Nijhawan(aasni448), Lakshidaa Saigiridharan (laksa656)

2019-12-07

Contents

1 Question 1: Principal components, including interpretation of them

Solve Exercise 8.18 of *Johnson, Wichern.* The data on the national track records for women, which you have studied earlier, can be found in the file `T1-9.dat`

a) Obtain the sample correlation matrix **R** for these data, and determine its eigenvalues and eigenvectors.

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

# Getting the sample correlation matrix and
# eigenvalues and egenvectors.
X = data[, 2:8]
X_corr = cor(X)
X_eigen = eigen(X_corr)


print("Sample Correlation Matrix")
```

```
## [1] "Sample Correlation Matrix"
```

```
print(X_corr)
```

```
##                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("Eigenvalues")
```

```
## [1] "Eigenvalues"
```

```
print(X_eigen$values)
```

```
## [1] 5.80762446 0.62869342 0.27933457 0.12455472 0.09097174 0.05451882
## [7] 0.01430226
```

```
print("Eigenvectors")
```

```
## [1] "Eigenvectors"
```

```
print(X_eigen$vectors)
```

```
##            [,1]       [,2]       [,3]        [,4]        [,5]        [,6]
## [1,] -0.3777657 -0.4071756 -0.1405803  0.58706293 -0.16706891 -0.53969730
## [2,] -0.3832103 -0.4136291 -0.1007833  0.19407501  0.09350016  0.74493139
## [3,] -0.3680361 -0.4593531  0.2370255 -0.64543118  0.32727328 -0.24009405
## [4,] -0.3947810  0.1612459  0.1475424 -0.29520804 -0.81905467  0.01650651
## [5,] -0.3892610  0.3090877 -0.4219855 -0.06669044  0.02613100  0.18898771
## [6,] -0.3760945  0.4231899 -0.4060627 -0.08015699  0.35169796 -0.24049968
## [7,] -0.3552031  0.3892153  0.7410610  0.32107640  0.24700821  0.04826992
##             [,7]
## [1,]  0.08893934
## [2,] -0.26565662
## [3,]  0.12660435
## [4,] -0.19521315
## [5,]  0.73076817
## [6,] -0.57150644
## [7,]  0.08208401
```

b) Determine the first two principal components for the standardized variables. Prepare a table showing the correlations of the standardized variables with the components, and the cumulative percentage of the total (standardized) sample variance explained by the two components.

```
Z = scale(X)
Z_corr = cor(Z)
```

```r
Z_eigen = eigen(Z_corr)

print("First principal component of the standardized variables.")
```

```
## [1] "First principal component of the standardized variables."
```

```r
print(Z_eigen$vectors[, 1])
```

```
## [1] -0.3777657 -0.3832103 -0.3680361 -0.3947810 -0.3892610 -0.3760945
## [7] -0.3552031
```

```r
print("Second principal component of the standardized variables.")
```

```
## [1] "Second principal component of the standardized variables."
```

```r
print(Z_eigen$vectors[, 2])
```

```
## [1] -0.4071756 -0.4136291 -0.4593531  0.1612459  0.3090877  0.4231899
## [7]  0.3892153
```

```r
print("Correlation of the standardized variables")
```

```
## [1] "Correlation of the standardized variables"
```

```r
print(Z_corr)
```

```
##                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
```

```r
print("Cumulative percentage of the total variance explained by the first two components")
```

```
## [1] "Cumulative percentage of the total variance explained by the first two components"
```

```r
print(sum(Z_eigen$values[1:2]) / 7)
```

```
## [1] 0.919474
```

```r
print("(91.9474%)")
```

```
## [1] "(91.9474%)"
```

c) Interpret the two principal components obtained in Part b. (Note that the first component is essentially a normalized unit vector and might measure the athletic excellence of a given nation. The second component might measure the relative strength of a nation at the various runnning distances.)

Most of the values of the first components are pretty close. In some sense, this component measures the average time on each of the tracks. So its an equally weighted performance measure.

The second component seems to be a measure of strenght regarding the distance of the runs. If the new component Y is positive, it means that nation better at shorter distances while if it's negative, it means that it performs better at longer distances.

d) Rank the nations based on their score on the first principal component. Does this ranking correspond with your intuitive notion of athletic excellence for the various countries?

```
Y_1 = as.matrix(Z) %*% Z_eigen$vectors[, 1]
rank = list(Country=data$Country, Score=Y_1)
rank = data.frame(rank)
ordered_idxs = order(rank$Score, decreasing=TRUE)
ordered_rank = rank[ordered_idxs, ]

print("Top 10 countries")
```

```
## [1] "Top 10 countries"
```

```
print(ordered_rank[1:10,])
```

```
##    Country    Score
## 54     USA 3.299149
## 18     GER 3.047517
## 45     RUS 3.042948
## 9      CHN 2.989467
## 17     FRA 2.518346
## 19     GBR 2.442706
## 13     CZE 2.406030
## 42     POL 2.273766
## 44     ROM 2.123006
## 2      AUS 1.931643
```

```
print("Bottom 10 countries")
```

```
## [1] "Bottom 10 countries"
```

```
print(ordered_rank[44:54,])
```

```
##    Country     Score
## 32     LUX -1.721468
## 23     INA -1.741942
## 34     MRI -1.749728
## 41     PHI -1.763534
## 12     CRC -2.166812
## 15     DOM -2.192410
## 47     SIN -3.093920
## 21     GUA -3.294124
## 40     PNG -5.257450
## 11     COK -7.906227
## 46     SAM -8.213415
```

This ranking makes sense since the countries on top are mostly developed nations who always perform well on sports while the ones at the bototm are underdeveloped nations that always lack performance on competitive sports.

2 Question 2: Factor Analysis

Task: Perform a factor analysis of the national track records for women given in Table 1.9. Use the sample covariance matrix S and interpret the factors. Compute factor scores, and check for outliers in the data. Repeat the analysis with the sample correlation matrix R. Does it make a difference if R, rather than S, is factored? Explain.

```
# ML Estimation
model_fact_S = factanal(data[,2:8], factors = 2, covmat = cov(X))
# PC
model_prin_S = principal(cov(data[,2:8]), nfactors = 2, covar = TRUE)

# ML Estimation
model_fact_R = factanal(data[,2:8], factors = 2, covmat = cor(X))
# PC
model_prin_R = principal(cor(data[,2:8]), nfactors = 2, covar = FALSE, scores = TRUE)
```

Question: What does it mean that the parameter rotation of factanal() is set to "varimax" by default

Answer: *In statistics, a varimax rotation is used to simplify the expression of a particular sub-space in terms of just a few major items each.* (from Wikipedia). It means that the rotation tries to fit as much variance as possible into as little components / features at possible. This has the benefit that first the new components might preserve some explainability and the found basis can actually used for dimensionality reduction as just a few features explain most the the variance. It is the default as this behaviour is what most people seek.

2.1 ML and S

2.1.1 Interpret the Factors

We can see that both components are explain approximately the same amount of variance. Looking at the values we see that factor1 puts more emphasis on longer distances (800m and more), while factor2 focuses mainly on shorter distances (400m and less). The model seems to be okay as both components explain around 89 percent of the variance.

```
model_fact_S
```

```
##
## Call:
## factanal(x = data[, 2:8], factors = 2, covmat = cov(X))
##
## Uniquenesses:
##      100      200      400      800     1500    3000m Marathon
##    0.094    0.024    0.152    0.144    0.016    0.028    0.338
##
## Loadings:
##          Factor1 Factor2
## 100      0.461   0.833
## 200      0.455   0.877
## 400      0.401   0.829
## 800      0.732   0.566
## 1500     0.882   0.454
## 3000m    0.918   0.361
## Marathon 0.693   0.427
##
##                Factor1 Factor2
```

```
## SS loadings      3.216   2.987
## Proportion Var   0.459   0.427
## Cumulative Var   0.459   0.886
##
## The degrees of freedom for the model is 8 and the fit was 0.6481
```

```
model_fact_S$loadings
```

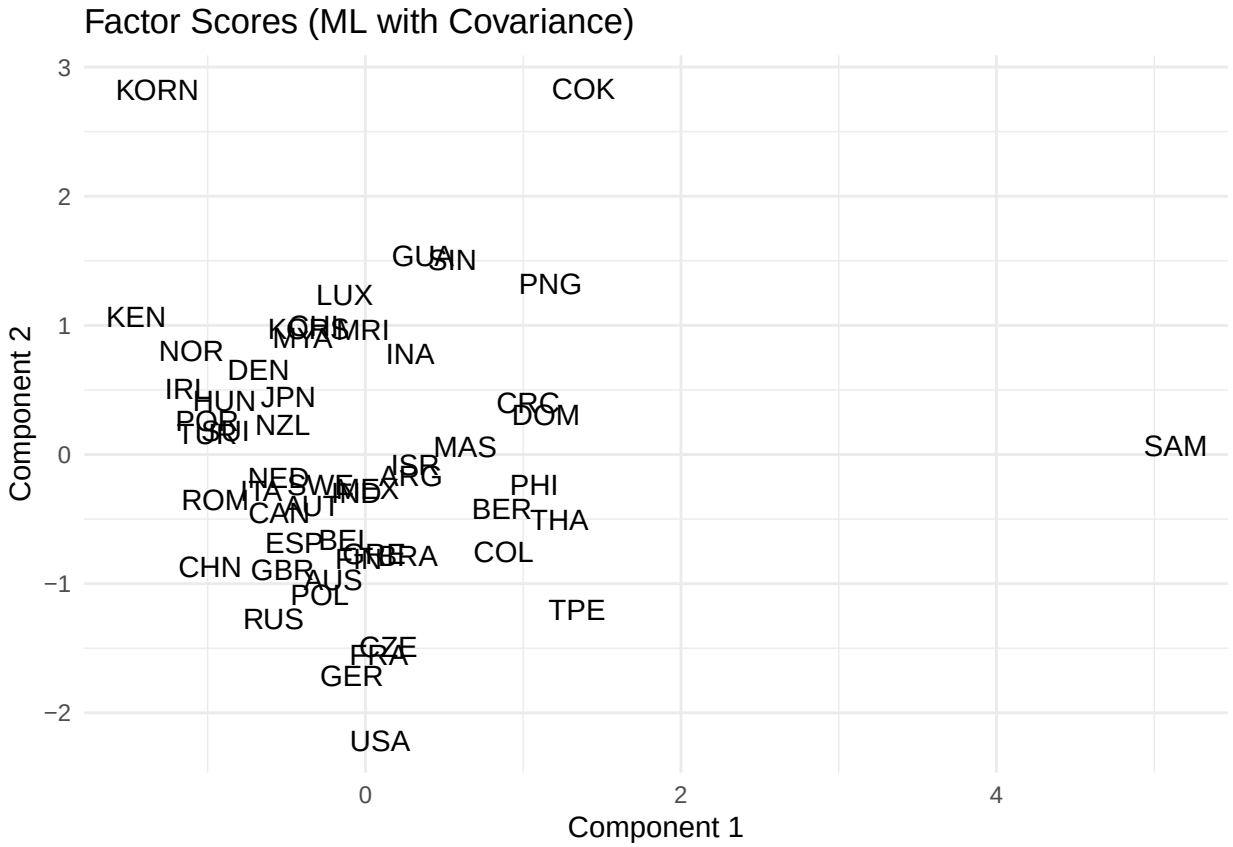
```
##
## Loadings:
##          Factor1 Factor2
## 100      0.461   0.833
## 200      0.455   0.877
## 400      0.401   0.829
## 800      0.732   0.566
## 1500     0.882   0.454
## 3000m    0.918   0.361
## Marathon 0.693   0.427
##
##                Factor1 Factor2
## SS loadings      3.216   2.987
## Proportion Var   0.459   0.427
## Cumulative Var   0.459   0.886
```

2.1.2 Compute the Factor Scores

```
model_fact_S_scores = factanal(data[,2:8], factors = 2, scores="Bartlett")$scores
```

2.1.3 Check for Outliers

We can see that the outliers are “SAM”, “KORN” and “COK”.



2.2 PC and S

2.2.1 Interpret the Factors

This time it seems that both factors focus quite a lot on the 400m track and the marathon. Component 2 puts a but more emphasis on the shorter tracks. So it seems like these components provide less interpretability. Looking at the difference we see that both components together explain less than 50 percent of the variance, which means that we lose quite a lot of the original variance. This also explains why it is quite hard to explain the components in terms of the given data - some part are missing.

```
model_prin_S
```

```
## Principal Components Analysis
## Call: principal(r = cov(data[, 2:8]), nfactors = 2, covar = TRUE)
## Unstandardized loadings (pattern matrix) based upon covariance matrix
##             RC1  RC2      h2      u2   H2      U2
## 100        0.17 0.31 1.2e-01 0.03100 0.80 2.0e-01
## 200        0.40 0.77 7.5e-01 0.11435 0.87 1.3e-01
## 400        1.04 2.38 6.7e+00 0.02014 1.00 3.0e-03
## 800        0.06 0.05 6.3e-03 0.00126 0.83 1.7e-01
## 1500       0.18 0.14 5.2e-02 0.02200 0.70 3.0e-01
## 3000m      0.56 0.37 4.5e-01 0.21213 0.68 3.2e-01
## Marathon 15.54 5.37 2.7e+02 0.00026 1.00 9.5e-07
##
##                         RC1   RC2
## SS loadings           243.00 35.37
```

```
## Proportion Var          0.87  0.13
## Cumulative Var          0.87  1.00
## Proportion Explained    0.87  0.13
## Cumulative Proportion   0.87  1.00
##
##  Standardized loadings (pattern matrix)
##          item  RC1  RC2   h2      u2
## 100         1 0.44 0.78 0.80 2.0e-01
## 200         2 0.43 0.82 0.87 1.3e-01
## 400         3 0.40 0.92 1.00 3.0e-03
## 800         4 0.70 0.58 0.83 1.7e-01
## 1500        5 0.66 0.52 0.70 3.0e-01
## 3000m       6 0.69 0.46 0.68 3.2e-01
## Marathon    7 0.95 0.33 1.00 9.5e-07
##
##                  RC1  RC2
## SS loadings     2.83 3.05
## Proportion Var  0.40 0.44
## Cumulative Var  0.40 0.84
## Cum. factor Var 0.48 1.00
##
## Mean item complexity =  1.6
## Test of the hypothesis that 2 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.02
##
## Fit based upon off diagonal values = 1
```

```
model_prin_S$loadings
```

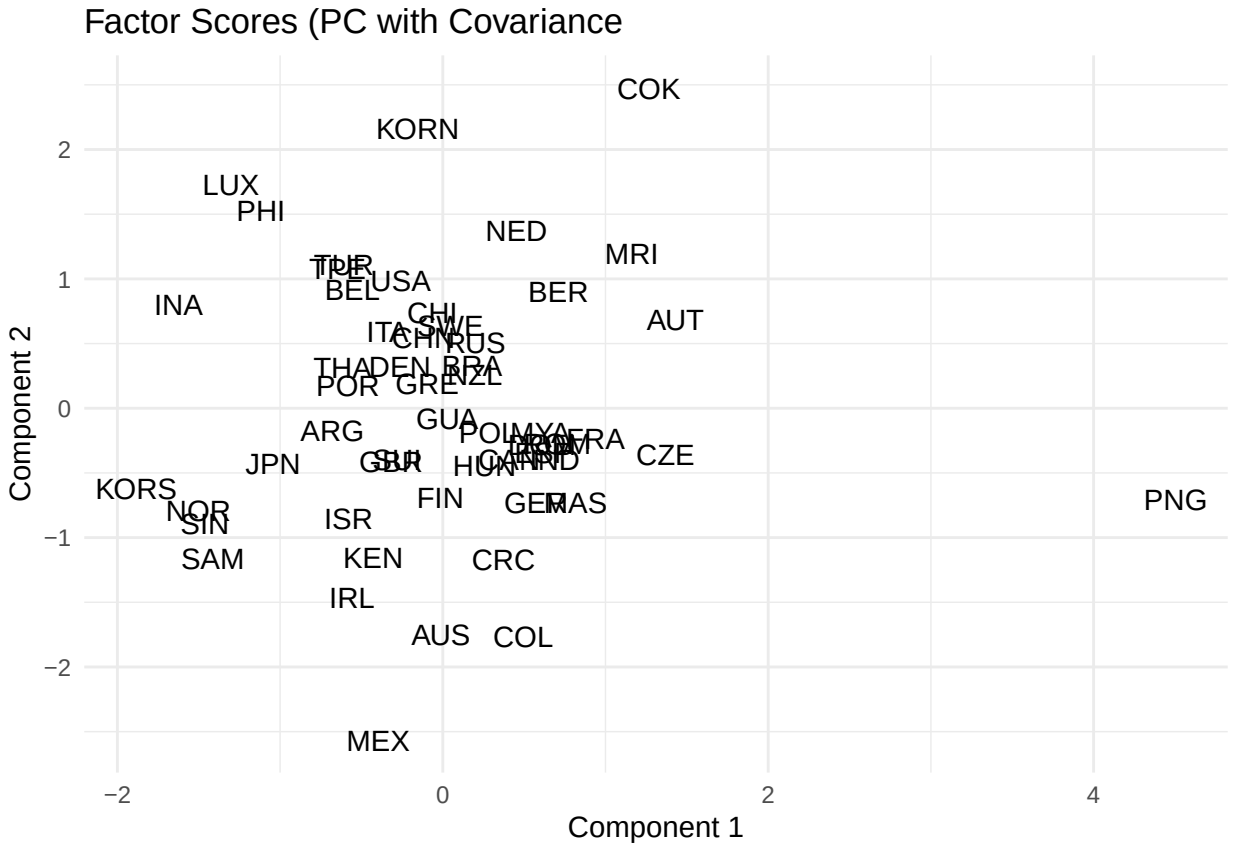
```
##
## Loadings:
##          RC1    RC2
## 100       0.173  0.307
## 200       0.404  0.765
## 400       1.038  2.376
## 800
## 1500      0.179  0.142
## 3000m     0.561  0.371
## Marathon 15.537  5.375
##
##                    RC1    RC2
## SS loadings    243.005 35.375
## Proportion Var  34.715  5.054
## Cumulative Var  34.715 39.768
```

2.2.2 Compute the Factor Scores

```
model_prin_S_scores = factor.scores(data[,2:8], f=model_prin_S)$scores
```

2.2.3 Check for Outliers

The outliers this time are “MEX”, “COK” and “PNG”.



2.3 ML and R

2.3.1 Interpretation of the Factors

Comparing with using the covariance matrix, we get the same explanation for the variables. Component 1 is still responsible for the longer tracks while component 2 is responsible for the shorter tracks. Both explain almost 90 percent of the variance which is quite good. Actually it seems that there is no difference at all. It seems there is n difference between using the covariance or correlation using the ML method.

```
model_fact_R
```

```
##
## Call:
## factanal(x = data[, 2:8], factors = 2, covmat = cor(X))
##
## Uniquenesses:
##      100      200      400      800     1500    3000m Marathon
##    0.094    0.024    0.152    0.144    0.016    0.028    0.338
##
## Loadings:
##          Factor1 Factor2
## 100      0.461   0.833
## 200      0.455   0.877
## 400      0.401   0.829
## 800      0.732   0.566
## 1500     0.882   0.454
```

```
## 3000m    0.918   0.361
## Marathon 0.693   0.427
##
##                Factor1 Factor2
## SS loadings      3.216   2.987
## Proportion Var   0.459   0.427
## Cumulative Var   0.459   0.886
##
## The degrees of freedom for the model is 8 and the fit was 0.6481
```

```
model_fact_R$loadings
```

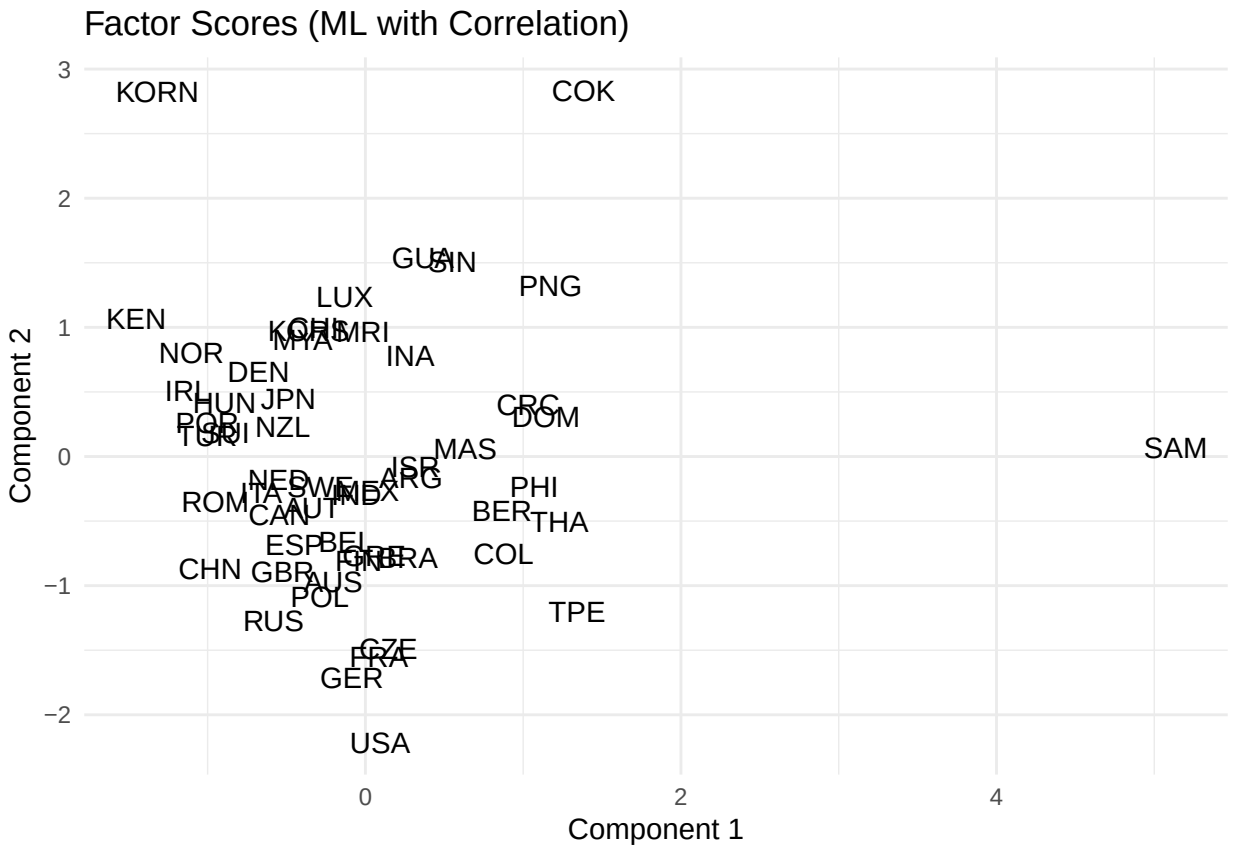
```
##
## Loadings:
##          Factor1 Factor2
## 100      0.461   0.833
## 200      0.455   0.877
## 400      0.401   0.829
## 800      0.732   0.566
## 1500     0.882   0.454
## 3000m    0.918   0.361
## Marathon 0.693   0.427
##
##                Factor1 Factor2
## SS loadings      3.216   2.987
## Proportion Var   0.459   0.427
## Cumulative Var   0.459   0.886
```

2.3.2 Compute the Factor Scores

```
model_fact_R_scores = factanal(data[,2:8], factors = 2, scores="Bartlett")$scores
```

2.3.3 Check for Outliers

We can see that the outliers are “SAM”, “KORN” and “COK”.



2.4 PC and R

2.4.1 Interpret the Factors

```
model_prin_R
```

```
## Principal Components Analysis
## Call: principal(r = cor(data[, 2:8]), nfactors = 2, covar = FALSE,
##     scores = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##           RC1  RC2   h2    u2 com
## 100      0.43 0.86 0.93 0.067 1.5
## 200      0.44 0.88 0.96 0.040 1.5
## 400      0.39 0.88 0.92 0.081 1.4
## 800      0.77 0.57 0.92 0.079 1.8
## 1500     0.85 0.48 0.94 0.060 1.6
## 3000m    0.89 0.39 0.93 0.066 1.4
## Marathon 0.83 0.37 0.83 0.172 1.4
##
##                        RC1  RC2
## SS loadings           3.31 3.13
## Proportion Var        0.47 0.45
## Cumulative Var        0.47 0.92
## Proportion Explained  0.51 0.49
## Cumulative Proportion 0.51 1.00
```

```
##
## Mean item complexity =  1.5
## Test of the hypothesis that 2 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.03
##
## Fit based upon off diagonal values = 1
```

```
model_prin_R$loadings
```

```
##
## Loadings:
##          RC1   RC2
## 100      0.431 0.865
## 200      0.437 0.877
## 400      0.385 0.878
## 800      0.773 0.569
## 1500     0.845 0.475
## 3000m    0.885 0.388
## Marathon 0.830 0.373
##
##                  RC1   RC2
## SS loadings    3.309 3.128
## Proportion Var 0.473 0.447
## Cumulative Var 0.473 0.919
```

We see that this time the values changed and it looks the we almost found the same components compared to the ML method in both cases (using S and R). Component 1 is again explaining the variance for the longer tracks and components to the variance for the shorter tracks. We see a slight improvement for the cumulative variance, as we actually explain more than 90 percent of the variance, which is quite good for just two components. So "just" these two explain almost all of the variance and both are good for interpretation.

2.4.2 Compute the Factor Scores

```
model_prin_R_scores = factor.scores(data[,2:8], f=model_prin_R)$scores
```

The outliers this time are "MEX", "COK" and "PNG", which are the same we observed with the above mentioned methods, which makes sense as we found almost the same components.

2.4.3 Check for Outliers