PH245 Homework 2 Solution

Xiangyu Hu, Toki Sherbakov

November 14, 2014

Rubrics: *Total: 20 pts*

Problem 1 (10 pts):

correlation	1pt
eigenvalues	1pt
eigenvectors	1pt
cumulative percentage	2pts
interpretation of PCs(loadings)	2pts
rankings	1pt
part(e)& (f) for completion	2pt

Problem 2 (10 pts):

covariance	1pt
using correlation/standardized variables	1pt
one-facor model loadings	1pt
one-facor model communalities	1pt
one-facor model proportion of variance	1pt
two-facor model loadings	1pt
two-facor model communalities	1pt
two-facor model proportion of variance	1pt
interpretation of rotated loadings	1pt*
After roatiaon: 2-factor model proportion of variance	1pt

* (there is no one single correct answer, as long as your interpretation makes sense)

Problem 1.

(a)

```
### PART A ###
# set work directory using setwd("The path you stored your data")
setwd("/Users/huxiangyu/Downloads/Archive")
# Import women's track dataset
women = read.delim("Data-HW3-track-women.dat", header = FALSE)

# Change the column names of the track women dataset
colnames(women) = c("Country", "100m", "200m", "400m", "800m", "1500m", "3000m", "Marathon")

#### PART A ####
# Subset out the first column since it contains the countries
women_vals = women[,-1]

# Find the correlation matrix R of the women's track dataset (remove 1st column because they are
# the countries)
R.women = cor(women_vals)
R.women

##            100m   200m   400m   800m  1500m  3000m Marathon
## 100m     1.0000 0.9411 0.8708 0.8092 0.7816 0.7279   0.6690
## 200m     0.9411 1.0000 0.9088 0.8198 0.8013 0.7319   0.6800
## 400m     0.8708 0.9088 1.0000 0.8058 0.7198 0.6738   0.6769
## 800m     0.8092 0.8198 0.8058 1.0000 0.9051 0.8666   0.8540
```

```
## 1500m    0.7816 0.8013 0.7198 0.9051 1.0000 0.9734   0.7906
## 3000m    0.7279 0.7319 0.6738 0.8666 0.9734 1.0000   0.7987
## Marathon 0.6690 0.6800 0.6769 0.8540 0.7906 0.7987   1.0000
```

```
# Determine eigenvalues/eigenvectors from R.women
## Method 1
eigen.women = eigen(R.women)
eigen.women
```

```
## $values
## [1] 5.80762 0.62869 0.27933 0.12455 0.09097 0.05452 0.01430
##
## $vectors
##         [,1]    [,2]    [,3]     [,4]     [,5]     [,6]     [,7]
## [1,] -0.3778 -0.4072 -0.1406  0.58706 -0.16707  0.53970  0.08894
## [2,] -0.3832 -0.4136 -0.1008  0.19408  0.09350 -0.74493 -0.26566
## [3,] -0.3680 -0.4594  0.2370 -0.64543  0.32727  0.24009  0.12660
## [4,] -0.3948  0.1612  0.1475 -0.29521 -0.81905 -0.01651 -0.19521
## [5,] -0.3893  0.3091 -0.4220 -0.06669  0.02613 -0.18899  0.73077
## [6,] -0.3761  0.4232 -0.4061 -0.08016  0.35170  0.24050 -0.57151
## [7,] -0.3552  0.3892  0.7411  0.32108  0.24701 -0.04827  0.08208
```

```
## Mehotd 2
pca.women = prcomp(women_vals, scale = TRUE)
# eigenvalues
(pca.women$sdev)^2
```

```
## [1] 5.80762 0.62869 0.27933 0.12455 0.09097 0.05452 0.01430
```

```
# eigenvectors (or loadings)
pca.women$rotation
```

```
##              PC1     PC2     PC3      PC4      PC5      PC6      PC7
## 100m     -0.3778  0.4072  0.1406 -0.58706  0.16707 -0.53970 -0.08894
## 200m     -0.3832  0.4136  0.1008 -0.19408 -0.09350  0.74493  0.26566
## 400m     -0.3680  0.4594 -0.2370  0.64543 -0.32727 -0.24009 -0.12660
## 800m     -0.3948 -0.1612 -0.1475  0.29521  0.81905  0.01651  0.19521
## 1500m    -0.3893 -0.3091  0.4220  0.06669 -0.02613  0.18899 -0.73077
## 3000m    -0.3761 -0.4232  0.4061  0.08016 -0.35170 -0.24050  0.57151
## Marathon -0.3552 -0.3892 -0.7411 -0.32108 -0.24701  0.04827 -0.08208
```

```
## Mehotd 3
pca.women2 = princomp(women_vals, cor = TRUE)
# eigenvalues
(pca.women2$sdev)^2
```

```
##  Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6  Comp.7
## 5.80762 0.62869 0.27933 0.12455 0.09097 0.05452 0.01430
```

```
# eigenvectors (or loadings)
pca.women2$loadings
```

```
##
## Loadings:
##          Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
## 100m     -0.378 -0.407 -0.141  0.587 -0.167 -0.540
## 200m     -0.383 -0.414 -0.101  0.194         0.745 -0.266
## 400m     -0.368 -0.459  0.237 -0.645  0.327 -0.240  0.127
## 800m     -0.395  0.161  0.148 -0.295 -0.819        -0.195
## 1500m    -0.389  0.309 -0.422                0.189  0.731
## 3000m    -0.376  0.423 -0.406         0.352 -0.240 -0.572
## Marathon -0.355  0.389  0.741  0.321  0.247
```

```
##
##                 Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
## SS loadings      1.000  1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.143  0.143  0.143  0.143  0.143  0.143  0.143
## Cumulative Var  0.143  0.286  0.429  0.571  0.714  0.857  1.000
```

```
#### PART B ####
# Determine the first 2 PCs of the standardized predictors
## method 1
women.std = scale(women_vals)
women.pc = women.std %*% eigen.women$vectors
head(women.pc[,1:2])
```

```
##         [,1]      [,2]
## [1,] -0.3932 -0.131611
## [2,]  1.9316  0.491067
## [3,]  1.2625  0.193148
## [4,]  1.2917 -0.002405
## [5,] -1.3961  0.760781
## [6,]  1.0068  0.379517
```

```
## method 2
pca.women = prcomp(women_vals, scale = TRUE)
head(pca.women$x[,1:2])
```

```
##          PC1       PC2
## [1,] -0.3932  0.131611
## [2,]  1.9316 -0.491067
## [3,]  1.2625 -0.193148
## [4,]  1.2917  0.002405
## [5,] -1.3961 -0.760781
## [6,]  1.0068 -0.379517
```

```
## method 3
pca.women2 = princomp(women_vals, cor=TRUE)
head(pca.women2$scores[,1:2])
```

```
##       Comp.1    Comp.2
## [1,] -0.3969 -0.132846
## [2,]  1.9498  0.495678
## [3,]  1.2744  0.194962
## [4,]  1.3039 -0.002428
## [5,] -1.4092  0.767924
## [6,]  1.0162  0.383081
```

```
# Find out the cumulative percentage of the total sample variance explained by the two components.
## method 1
cumsum(eigen.women$values)/sum(eigen.women$values)
```

```
## [1] 0.8297 0.9195 0.9594 0.9772 0.9902 0.9980 1.0000
```

```
## method 2
summary(pca.women)
```

```
## Importance of components:
##                         PC1    PC2    PC3    PC4   PC5     PC6     PC7
## Standard deviation     2.41 0.7929 0.5285 0.3529 0.302 0.23349 0.11959
## Proportion of Variance 0.83 0.0898 0.0399 0.0178 0.013 0.00779 0.00204
## Cumulative Proportion  0.83 0.9195 0.9594 0.9772 0.990 0.99796 1.00000
```

```
## method 3
summary(pca.women2)
```

```
## Importance of components:
##                           Comp.1  Comp.2 Comp.3  Comp.4 Comp.5   Comp.6
## Standard deviation     2.4099 0.79290 0.5285 0.35292 0.3016 0.233493
## Proportion of Variance 0.8297 0.08981 0.0399 0.01779 0.0130 0.007788
## Cumulative Proportion  0.8297 0.91947 0.9594 0.97717 0.9902 0.997957
##                          Comp.7
## Standard deviation     0.119592
## Proportion of Variance 0.002043
## Cumulative Proportion  1.000000
```

We see that the first PC explains 82.97% of the variance in the data and second PC explains 8.98% of the variance in the data. The cumulative percentage of the total sample variance explained by the 2 PCs is 91.95%.

(c) *There is a little confusion here for some of you. It meant to interpret the principal component loadings*

```
## method 1
eigen.women$vectors[,1:2]
```

```
##         [,1]    [,2]
## [1,] -0.3778 -0.4072
## [2,] -0.3832 -0.4136
## [3,] -0.3680 -0.4594
## [4,] -0.3948  0.1612
## [5,] -0.3893  0.3091
## [6,] -0.3761  0.4232
## [7,] -0.3552  0.3892
```

```
## method 2
pca.women$rotation[,1:2]
```

```
##              PC1     PC2
## 100m     -0.3778  0.4072
## 200m     -0.3832  0.4136
## 400m     -0.3680  0.4594
## 800m     -0.3948 -0.1612
## 1500m    -0.3893 -0.3091
## 3000m    -0.3761 -0.4232
## Marathon -0.3552 -0.3892
```

```
## method 3
pca.women2$loadings[,1:2]
```

```
##           Comp.1  Comp.2
## 100m     -0.3778 -0.4072
## 200m     -0.3832 -0.4136
## 400m     -0.3680 -0.4594
## 800m     -0.3948  0.1612
## 1500m    -0.3893  0.3091
## 3000m    -0.3761  0.4232
## Marathon -0.3552  0.3892
```

All track events contribute about equally to the first principal component. This component might be called a track index or track excellence component. The second component contrasts the times for the shorter distances (100m, 200m, 400m) with the times for the longer distances (800m, 1500m, 3000m, marathon) and might be called a distance component.

(d)

```
# The "track excellence" rankings for the 54 countries are taken from ordering the scores of
# the first PC. We need to make the order decreasing so that we have the highest score first,
# and the lowest score last.
rankings.women = women[order(pca.women$x[,1], decreasing = TRUE), 1]
```

```r
head(rankings.women)

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

The data are measured in time, since the coefficients for the first component are all negative. The bigger the score of the first component, the better the performance.
These rankings appear to be consistent with intuitive notions of athletic excellence.
You may hold a different opinion thinking that African countries should performs the best. (It is totally OK, no points deducted for this)
(e)

```r
# Need to convert the values into meters per second

# First, convert the last 4 columns into seconds (they were originally measured in minutes)
women_mpers = women_vals
women_mpers[,4:7] = women_mpers[,4:7]*60

# Next, inverse the values so that each value is per second
women_mpers = 1/women_mpers

# Multiply each column by their respective distances
for(i in 1:ncol(women_mpers)){

  # If we are on the last column (marathon), set the track.dist to 42195
  if(i == ncol(women_mpers)){track.dist = 42195}

  # Take the distance from the column name
  else{track.dist = as.numeric(strsplit(colnames(women_mpers)[i], "m"))}

  # Multiply the current column by its distance
  women_mpers[,i] = women_mpers[,i] * track.dist

}

# Perform PCA using covariance matrix (no scaling needed)
pca.women.mpers = prcomp(women_mpers)
pca.women.mpers

## Standard deviations:
## [1] 0.85566 0.29338 0.18270 0.12238 0.09408 0.07853 0.04545
##
## Rotation:
##             PC1      PC2      PC3      PC4      PC5      PC6      PC7
## 100m     0.3102 -0.37597 -0.09756  0.58480 -0.04613  0.62433 -0.13776
## 200m     0.3574 -0.43377 -0.08896  0.32288 -0.02978 -0.68871  0.31104
## 400m     0.3787 -0.51873  0.27425 -0.66667 -0.18727  0.12377 -0.13199
## 800m     0.2993  0.05314  0.05252 -0.12809  0.89434  0.13592  0.26473
## 1500m    0.3912  0.21084 -0.43499 -0.05511  0.12725 -0.23626 -0.73364
## 3000m    0.4596  0.39557 -0.42664 -0.18389 -0.35674  0.19926  0.49949
## Marathon 0.4227  0.44458  0.73032  0.23676 -0.13640 -0.08106 -0.09516

summary(pca.women.mpers)

## Importance of components:
##                          PC1    PC2    PC3    PC4    PC5     PC6     PC7
## Standard deviation     0.856 0.2934 0.1827 0.1224 0.0941 0.07853 0.04545
## Proportion of Variance 0.829 0.0974 0.0378 0.0169 0.0100 0.00698 0.00234
## Cumulative Proportion  0.829 0.9259 0.9637 0.9807 0.9907 0.99766 1.00000
```

```
rankings.women.mpers = women[order(pca.women.mpers$x[,1], decreasing = TRUE), 1]
rankings.women.mpers

##  [1] USA    CHN    RUS    GER    GBR    FRA    ROM    POL    CZE    AUS
## [11] ESP    CAN    ITA    NED    IRL    POR    KEN    FIN    BEL    SUI
## [21] MEX    AUT    GRE    TUR    HUN    NOR    BRA    NZL    SWE    JPN
## [31] DEN    IND    COL    ARG    KOR, S ISR    MYA    CHI    TPE    KOR, N
## [41] LUX    MAS    THA    INA    BER    MRI    PHI    CRC    DOM    SIN
## [51] GUA    PNG    COK    SAM
## 54 Levels: ARG AUS AUT BEL BER BRA CAN CHI CHN COK COL CRC CZE DEN ... USA
```

The cumulative percentage of total sample variance explained by the first 2 PCs is 92.59%. The interpretation of the sample component is similar to the interpretation in part (b). All track events contribute about equally to the first component. This component might be called a track index or track excellence component. The second component contrasts times in m/s for the shorter distances (100m, 200m, 400m) with the times for the longer distances (800m, 1500m, 3000m, Marathon) and might be called the distance component.

The "track excellence" rankings for the countries are very similar to the rankings for the countrie obtained in part (d). They only slightly differ. I would prefer the second method because the first 2 PCs explain slightly more of the total variance than the first method. Also, the second method has data in consistent units. We know that PCA is sensitive to the scale of the data itself, so making our units consistent is also a good operation.

(f) *To save some time, we didn't show all three methods...*

```
# Import track dataset for men
men = read.table("Data-HW3-track-men.dat", quote="\"")

# Change the column names of the track men dataset
colnames(men) = c("Country", "100m", "200m", "400m", "800m", "1500m", "5000m", "10000m", "Marathon")

# Extract just the values (not the countries)
men_vals = men[,-1]

# Obtain correlation matrix R
R.men = cor(men_vals)
R.men

##            100m   200m   400m   800m  1500m  5000m 10000m Marathon
## 100m     1.0000 0.9148 0.8041 0.7119 0.7658 0.7399 0.7148   0.6765
## 200m     0.9148 1.0000 0.8449 0.7969 0.7951 0.7613 0.7480   0.7211
## 400m     0.8041 0.8449 1.0000 0.7677 0.7716 0.7797 0.7657   0.7127
## 800m     0.7119 0.7969 0.7677 1.0000 0.8958 0.8607 0.8431   0.8070
## 1500m    0.7658 0.7951 0.7716 0.8958 1.0000 0.9165 0.9013   0.8778
## 5000m    0.7399 0.7613 0.7797 0.8607 0.9165 1.0000 0.9882   0.9441
## 10000m   0.7148 0.7480 0.7657 0.8431 0.9013 0.9882 1.0000   0.9542
## Marathon 0.6765 0.7211 0.7127 0.8070 0.8778 0.9441 0.9542   1.0000

# Obtain eigenvalues and eigenvectors of R.men
eigen.men = eigen(R.men)
eigen.men

## $values
## [1] 6.703290 0.638410 0.227524 0.205849 0.097577 0.070688 0.046942 0.009719
##
## $vectors
##         [,1]     [,2]      [,3]     [,4]     [,5]     [,6]    [,7]
## [1,] -0.3324 -0.52940 -0.343859  0.38075  0.29967 -0.36204  0.3476
## [2,] -0.3461 -0.47039  0.003786  0.21702 -0.54143  0.34859 -0.4399
## [3,] -0.3391 -0.34533  0.067061 -0.85130  0.13299  0.07708  0.1136
## [4,] -0.3530  0.08946  0.782711  0.13428 -0.22728 -0.34131  0.2589
## [5,] -0.3660  0.15365  0.244270  0.23302  0.65162  0.52978 -0.1470
## [6,] -0.3698  0.29476 -0.182863 -0.05462  0.07182 -0.35914 -0.3283
```

```
## [7,] -0.3659  0.33361 -0.243981 -0.08707 -0.06133 -0.27309 -0.3511
## [8,] -0.3543  0.38656 -0.334633  0.01812 -0.33789  0.37517  0.5942
##          [,8]
## [1,] -0.06570
## [2,]  0.06076
## [3,] -0.00347
## [4,] -0.03927
## [5,] -0.03975
## [6,]  0.70568
## [7,] -0.69718
## [8,]  0.06932
```

```
# Obtain PCA with standardizing the variables
pca.men = prcomp(men_vals, scale = TRUE)
pca.men
```

```
## Standard deviations:
## [1] 2.58907 0.79901 0.47700 0.45371 0.31237 0.26587 0.21666 0.09858
##
## Rotation:
##              PC1      PC2       PC3      PC4      PC5      PC6     PC7
## 100m     -0.3324 -0.52940 -0.343859 -0.38075  0.29967 -0.36204  0.3476
## 200m     -0.3461 -0.47039  0.003786 -0.21702 -0.54143  0.34859 -0.4399
## 400m     -0.3391 -0.34533  0.067061  0.85130  0.13299  0.07708  0.1136
## 800m     -0.3530  0.08946  0.782711 -0.13428 -0.22728 -0.34131  0.2589
## 1500m    -0.3660  0.15365  0.244270 -0.23302  0.65162  0.52978 -0.1470
## 5000m    -0.3698  0.29476 -0.182863  0.05462  0.07182 -0.35914 -0.3283
## 10000m   -0.3659  0.33361 -0.243981  0.08707 -0.06133 -0.27309 -0.3511
## Marathon -0.3543  0.38656 -0.334633 -0.01812 -0.33789  0.37517  0.5942
##               PC8
## 100m     -0.06570
## 200m      0.06076
## 400m     -0.00347
## 800m     -0.03927
## 1500m    -0.03975
## 5000m     0.70568
## 10000m   -0.69718
## Marathon  0.06932
```

```
summary(pca.men)
```

```
## Importance of components:
##                          PC1    PC2    PC3    PC4    PC5     PC6     PC7
## Standard deviation     2.589 0.7990 0.4770 0.4537 0.3124 0.26587 0.21666
## Proportion of Variance 0.838 0.0798 0.0284 0.0257 0.0122 0.00884 0.00587
## Cumulative Proportion  0.838 0.9177 0.9462 0.9719 0.9841 0.99292 0.99879
##                            PC8
## Standard deviation     0.09858
## Proportion of Variance 0.00121
## Cumulative Proportion  1.00000
```

```
# ranking
rankings.men = men[order(pca.men$x[,1], decreasing = TRUE), 1]
rankings.men
```

```
##  [1] U.S.A.         GreatBritain   Kenya          France
##  [5] Australia      Italy          Brazil         Germany
##  [9] Portugal       Canada         Belgium        Poland
## [13] Russia         Spain          Japan          Switzerland
## [17] Norway         Netherlands    Mexico         NewZealand
## [21] Denmark        Greece         Hungary        Finland
```

```
## [25] Ireland        Sweden         Austria        Chile
## [29] China          CzechRepublic  Romania        Argentina
## [33] Korea,South    India          Columbia       Turkey
## [37] Israel         Mauritius      Luxembourg     Taiwan
## [41] DominicanRepub Bermuda        Thailand       Indonesia
## [45] CostaRica      Korea,North    Malaysia       Guatemala
## [49] Philippines    Myanmar(Burma) PapuaNewGuinea Singapore
## [53] Samoa          CookIslands
## 54 Levels: Argentina Australia Austria Belgium Bermuda Brazil ... U.S.A.
```

The cumulative percentage of total sample variance explained by the first 2 PCs is 91.77%.All track events contribute about equally to the first component. This component might be called a track index or track excellence component. The second component contrasts the times for the shorter distances (100m, 200m, 400m) with the times for the longer distances (800m, 1500m, 5000m, 10000m, Marathon) and might be called a distance component. For the ranking, again U.S.A ranks on the top.
The PCA of the men's track data is consistent with that of the women.
Problem 2. (a)-(c)

```
# Import the air pollution dataset
pollution = read.table("Data-HW3-pollution.dat", quote="\"")

# Change the column names of pollution dataset
colnames(pollution) = c("Wind", "SolarRad", "CO", "NO", "NO2", "O3", "HC")

#### PART A ####

# Generate sample covariance matrix for air pollution dataset
S.pol = cov(pollution)
S.pol

##             Wind SolarRad      CO      NO     NO2      O3     HC
## Wind      2.5000  -2.7805 -0.3780 -0.4634 -0.5854 -2.2317 0.1707
## SolarRad -2.7805 300.5157  3.9094 -1.3868  6.7631 30.7909 0.6237
## CO       -0.3780   3.9094  1.5221  0.6736  2.3148  2.8217 0.1417
## NO       -0.4634  -1.3868  0.6736  1.1823  1.0883 -0.8107 0.1765
## NO2      -0.5854   6.7631  2.3148  1.0883 11.3635  3.1266 1.0441
## O3       -2.2317  30.7909  2.8217 -0.8107  3.1266 30.9785 0.5947
## HC        0.1707   0.6237  0.1417  0.1765  1.0441  0.5947 0.4785

# Notice that SolarRadiation, O3, and NO2 have relatively large variances compared to the other
# variables. Due to this, it is necessary to scale the variables and perform factor analysis.

#### PART B ####

# Obtain PCA solution
pol_pca = prcomp(pollution, scale = TRUE)

# Obtain square root of eigenvalues (used to calculate factor loadings)
eigenvalues_sqrt = pol_pca$sdev

# Eigenvectors: a.k.a loadings (used to calculate factor loadings)
pol_loadings = pol_pca$rotation

# Obtain factor model with m = 1
L1 = cbind(pol_loadings[,1] * eigenvalues_sqrt[1])
L1

##              [,1]
## Wind     -0.3620
```

```
## SolarRad  0.3142
## CO        0.8424
## NO        0.5772
## NO2       0.7613
## O3        0.4961
## HC        0.4883

# Obtain factor model with m = 2
L2 = cbind(pol_loadings[,1] * eigenvalues_sqrt[1], pol_loadings[,2] * eigenvalues_sqrt[2])
L2

##              [,1]      [,2]
## Wind     -0.3620  0.327809
## SolarRad  0.3142 -0.619975
## CO        0.8424 -0.008028
## NO        0.5772  0.511736
## NO2       0.7613  0.235183
## O3        0.4961 -0.667490
## HC        0.4883  0.362466

# Find commonalities for m = 1 factor model
h1 = apply(L1^2, 1, sum)
h1

##     Wind SolarRad       CO       NO      NO2       O3       HC
##  0.13106  0.09875  0.70967  0.33321  0.57957  0.24614  0.23839

# Find commonalities for m = 2 factor model
h2 = apply(L2^2, 1, sum)
h2

##     Wind SolarRad       CO       NO      NO2       O3       HC
##   0.2385   0.4831   0.7097   0.5951   0.6349   0.6917   0.3698

#### PART C ####

# Find proportion of variation accounted for by the m = 1 factor model
var.exp.m1.f1 = sum(L1[,1]^2)/7
var.exp.m1.f1

## [1] 0.3338

# Find proportion of variation accounted for by the m = 2 factor model
var.exp.m2.f1 = sum(L2[,1]^2)/7
var.exp.m2.f2 = sum(L2[,2]^2)/7
var.exp.m2.f1 + var.exp.m2.f2

## [1] 0.5318

# We see that 33.38% of the variance is accounted for by the m = 1 factor model and
# 53.18% of the variance is accounted for by the m = 2 factor model.
```

(d)

```
# Perform varimax rotation on m = 2 factor model
L2_rot = varimax(L2, normalize = FALSE)
L2_rot

## $loadings
##
## Loadings:
```

```
##          [,1]   [,2]
## Wind     -0.160  0.461
## SolarRad         -0.695
## CO        0.735 -0.412
## NO        0.752  0.171
## NO2       0.781 -0.160
## O3        0.114 -0.824
## HC        0.602
##
##                 [,1]  [,2]
## SS loadings    2.117 1.606
## Proportion Var 0.302 0.229
## Cumulative Var 0.302 0.532
##
## $rotmat
##        [,1]    [,2]
## [1,] 0.8768 -0.4808
## [2,] 0.4808  0.8768
```

It seems that in the first factor, there would be a grouping with CO, NO, NO2, and HC due to their high loadings. Also, there appears to be a contrast between Wind and the other variables. The second factor is harder to interpret but there appears to be higher loadings on SolarRadiation and Ozone, maybe hinting that those two should group together as well. In factor 2, there is a contrast between Wind and NO with SolarRadiation, CO, NO2, and O3.

After rotation, 53.2% of the variation is accounted for by the m = 2 factor model.