Zihao Huang- Homework4

Zihao Huang

April 30, 2018

```
#STAT223 Homework 4
#Zihao Huang
#4/27/2018


#1.
library(readxl)
```

```
## Warning: package 'readxl' was built under R version 3.4.3
```

```
track <- read_excel("C:/Users/simon/Desktop/STAT223/track.xlsx")
track
```

```
## # A tibble: 55 x 9
##      Country `100m` `200m` `400m` `800m` `1500m` `5000m` `10,00m` Marathon
##        <chr>  <dbl>  <dbl>  <dbl>  <dbl>   <dbl>   <dbl>    <dbl>    <dbl>
##  1 argentina  10.39  20.81  46.84   1.81    3.70   14.04    29.36   137.72
##  2 australia  10.31  20.06  44.84   1.74    3.57   13.28    27.66   128.30
##  3   austria  10.44  20.81  46.82   1.79    3.60   13.26    27.72   135.90
##  4   belgium  10.34  20.68  45.04   1.73    3.60   13.22    27.45   129.95
##  5   bermuda  10.28  20.58  45.91   1.80    3.75   14.68    30.55   146.62
##  6    brazil  10.22  20.43  45.21   1.73    3.66   13.62    28.62   133.13
##  7     burma  10.64  21.52  48.30   1.80    3.85   14.45    30.28   139.95
##  8    canada  10.17  20.22  45.68   1.76    3.63   13.55    28.09   130.15
##  9     chile  10.34  20.80  46.20   1.79    3.71   13.61    29.30   134.03
## 10     china  10.51  21.04  47.30   1.81    3.73   13.90    29.13   133.53
## # ... with 45 more rows
```

```
tra<-track[,2:9]
S<-var(tra)
R<-cor(tra)
n<-nrow(tra)
p<-ncol(tra)
##a.
S
```

```
##                100m       200m       400m        800m      1500m
## 100m     0.12350249 0.20902182 0.43069956 0.016920438 0.03836684
## 200m     0.20902182 0.41557024 0.79905603 0.033115455 0.07788771
## 400m     0.43069956 0.79905603 2.12290020 0.080743131 0.18974209
## 800m     0.01692044 0.03311545 0.08074313 0.004055758 0.00911532
## 1500m    0.03836684 0.07788771 0.18974209 0.009115320 0.02430774
## 5000m    0.17441020 0.35913859 0.90887976 0.044062088 0.11592929
## 10,00m   0.40184545 0.81171145 2.07341549 0.100049327 0.26343721
## Marathon 1.68601222 3.54620963 9.47785704 0.473903333 1.24516296
##               5000m     10,00m   Marathon
## 100m     0.17441020  0.4018455  1.6860122
## 200m     0.35913859  0.8117114  3.5462096
## 400m     0.90887976  2.0734155  9.4778570
```

```
## 800m     0.04406209  0.1000493  0.4739033
## 1500m    0.11592929  0.2634372  1.2451630
## 5000m    0.64185811  1.4115480  6.8910485
## 10,00m   1.41154798  3.2678936 15.7321815
## Marathon 6.89104852 15.7321815 85.1381467
```

```
(eval<-eigen(S)$values)
```

```
## [1] 8.991362e+01 1.412626e+00 2.598442e-01 1.094203e-01 2.730060e-02
## [6] 1.273280e-02 2.243554e-03 4.455645e-04
```

```
evec<-eigen(S)$vectors
##b.
percentage<-rep(0,8)
for (i in 1:8){
  percentage[i] <- sum(eval[1:i])/sum(diag(eval))
}
percentage
```

```
## [1] 0.9801107 0.9955091 0.9983416 0.9995343 0.9998319 0.9999707 0.9999951
## [8] 1.0000000
```

```
length(eval[eval>mean(eval)])
```

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

```
###It recommends 1 PCs.
sum(eval[1])/sum(eval)
```

```
## [1] 0.9801107
```

```
###The proportion of total variance is 98.01%.
##c.
z <- as.matrix(scale(tra,center=T, scale=F))%*%evec[,1:2]
rbind(colnames(track)[-1],t(evec[,1:2]))
```

```
##      [,1]                  [,2]                  [,3]
## [1,] "100m"                "200m"                "400m"
## [2,] "-0.0198654068009731" "-0.0415544989050151" "-0.110631838371887"
## [3,] "-0.210689575996937"  "-0.358925791280443"  "-0.827862507500363"
##      [,4]                   [,5]                   [,6]
## [1,] "800m"                 "1500m"                "5000m"
## [2,] "-0.00548769939006153" "-0.0143868222115944" "-0.079308443877455"
## [3,] "-0.023174899381864"   "-0.044652546578267"  "-0.129961339234901"
##      [,7]                 [,8]
## [1,] "10,00m"             "Marathon"
## [2,] "-0.18109899442057"  "-0.972787445963142"
## [3,] "-0.298853928879327" "0.180807359406435"
```

```
###The first principal component explains the long run ability (Marathon). The second
###principal component explains the 400m running ability.
##d.
z1.1<-cbind(as.numeric(z[,1]),track[,1])
best5.1<-z1.1[order(z1.1[,1],decreasing = T),c(1,2)][1:5,]
worst5.1<-z1.1[order(z1.1[,1],decreasing = F),c(1,2)][1:5,]
###Best 5 are Usa,Australia,Japan,Portugal,Netherla.
###(The nations need the shortest time to finish running)
###Worst 5 are Cookis,Wsamoa,Singapor,Domrep,Malaysia.
```

```
###(The nations need the longest time to finish running)


#2.
##a.
R
```

```
##               100m      200m      400m      800m     1500m     5000m
## 100m     1.0000000 0.9226384 0.8411468 0.7560278 0.7002382 0.6194618
## 200m     0.9226384 1.0000000 0.8507270 0.8066265 0.7749513 0.6953770
## 400m     0.8411468 0.8507270 1.0000000 0.8701714 0.8352694 0.7786139
## 800m     0.7560278 0.8066265 0.8701714 1.0000000 0.9180442 0.8635939
## 1500m    0.7002382 0.7749513 0.8352694 0.9180442 1.0000000 0.9281140
## 5000m    0.6194618 0.6953770 0.7786139 0.8635939 0.9281140 1.0000000
## 10,00m   0.6325389 0.6965391 0.7872045 0.8690489 0.9346970 0.9746354
## Marathon 0.5199490 0.5961837 0.7049905 0.8064764 0.8655492 0.9321884
##             10,00m  Marathon
## 100m     0.6325389 0.5199490
## 200m     0.6965391 0.5961837
## 400m     0.7872045 0.7049905
## 800m     0.8690489 0.8064764
## 1500m    0.9346970 0.8655492
## 5000m    0.9746354 0.9321884
## 10,00m   1.0000000 0.9431763
## Marathon 0.9431763 1.0000000
```

```
(eval.2<-eigen(R)$values)
```

```
## [1] 6.62214613 0.87761829 0.15932114 0.12404939 0.07988027 0.06796515
## [7] 0.04641953 0.02260010
```

```
evec.2<-eigen(R)$vectors
##b.
percentage<-rep(0,8)
for (i in 1:8){
  percentage[i] <- sum(eval.2[1:i])/sum(diag(eval.2))
}
percentage
```

```
## [1] 0.8277683 0.9374706 0.9573857 0.9728919 0.9828769 0.9913725 0.9971750
## [8] 1.0000000
```

```
length(eval.2[eval.2>mean(eval.2)])
```

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

```
###It recommends 1 PCs.
sum(eval.2[1])/sum(eval.2)
```

```
## [1] 0.8277683
```

```
###The proportion of total variance is 82.78%.
##c.
#z.2<-as.matrix(scale(tra,center=T, scale=F))%*%evec.2[,1:2]
tra.st<-matrix(0,ncol=8,nrow=55)
tra<-as.matrix(tra)
for (i in 1:55){
```

```
  for (j in 1:8){
    tra.st[i,j]<-(tra[i,j]-colMeans(tra)[j])/sqrt(diag(S)[j])
  }
}
z.2<-tra.st%*%evec.2[,1]
evec.2[,1:2]
```

```
##            [,1]        [,2]
## [1,] -0.3175565 -0.56687750
## [2,] -0.3369792 -0.46162589
## [3,] -0.3556454 -0.24827331
## [4,] -0.3686841 -0.01242993
## [5,] -0.3728099  0.13979665
## [6,] -0.3643741  0.31203045
## [7,] -0.3667726  0.30685985
## [8,] -0.3419261  0.43896267
```

```
###The first principal component doesn't interpret well since all of the values
###are smaller than -0.40,
###The second principal interpret the short run ability (100m running)
##d.
z1.2<-cbind(as.numeric(z.2[,1]),track[,1])
best5.2<-z1.2[order(z1.2[,1],decreasing = T),c(1,2)][1:5,]
worst5.2<-z1.2[order(z1.2[,1],decreasing = F),c(1,2)][1:5,]
###Best 5 are Usa,gbni,italy,Ussr,gdr.
###(The nations need the shortest time to finish running)
###Worst 5 are Cookis,Wsamoa,Mauritiu,png,Singapor.
###(The nations need the longest time to finish running)
##e.
(cbind(best5.1,best5.2))
```

```
##    as.numeric(z[, 1])   Country as.numeric(z.2[, 1]) Country
## 53           8.857144       usa             3.430556     usa
## 2            8.601844 australia             3.024230    gbni
## 30           8.113760     japan             2.726950   italy
## 44           8.066707  portugal             2.626851    ussr
## 38           7.833508  netherla             2.590092     gdr
```

```
###From the result on the best 5 nations between 1 and 2, Usa is the top 2 nations.
(cbind(worst5.1,worst5.2))
```

```
##    as.numeric(z[, 1])  Country as.numeric(z.2[, 1])  Country
## 12          -29.55186   cookis           -10.555626   cookis
## 55          -26.08673   wsamoa            -7.231216   wsamoa
## 46          -21.21484 singapor            -4.258658 mauritiu
## 16          -17.57318   domrep            -3.909193      png
## 35          -17.41341 malaysia            -3.122111 singapor
```

```
###From the result on the worst 5 nations between 1 and 2, Cookis, Wsamoa, Singapor  are on the list.
```

```
#3.
##a
tra3<-cbind(tra[,1:3],60*tra[,4:8])
tramps<-cbind(100/tra3[,1],200/tra3[,2],400/tra3[,3],800/tra3[,4],1500/tra3[,5],
              5000/tra3[,6],10000/tra3[,7],42195/tra3[,8])
tramps<-as.matrix(tramps)
```

```
##b
(S3<-var(tramps))
```

```
##            [,1]       [,2]       [,3]       [,4]       [,5]       [,6]
## [1,] 0.09044592 0.07985342 0.06436012 0.05645566 0.05575291 0.05787011
## [2,] 0.07985342 0.08272329 0.06308333 0.05785661 0.05966131 0.06364612
## [3,] 0.06436012 0.06308333 0.06698808 0.05650067 0.05789093 0.06427206
## [4,] 0.05645566 0.05785661 0.05650067 0.06272468 0.06116187 0.06898807
## [5,] 0.05575291 0.05966131 0.05789093 0.06116187 0.07216120 0.08024026
## [6,] 0.05787011 0.06364612 0.06427206 0.06898807 0.08024026 0.10417999
## [7,] 0.06050769 0.06514185 0.06648693 0.07075964 0.08239365 0.10338860
## [8,] 0.04822916 0.05397306 0.05815799 0.06376408 0.07389448 0.09565101
##            [,7]       [,8]
## [1,] 0.06050769 0.04822916
## [2,] 0.06514185 0.05397306
## [3,] 0.06648693 0.05815799
## [4,] 0.07075964 0.06376408
## [5,] 0.08239365 0.07389448
## [6,] 0.10338860 0.09565101
## [7,] 0.10862125 0.09919269
## [8,] 0.09919269 0.10214340
```

```
(eval.3<-eigen(S3)$values)
```

```
## [1] 0.565471242 0.083396555 0.012371382 0.009348032 0.006793014 0.005854749
## [7] 0.004160540 0.002592306
```

```
evec.3<-eigen(S3)$vectors
##c.
length(eval.3[eval.3>mean(eval.3)])
```

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

```
###It recommends 1 PCs.
sum(eval.3[1])/sum(eval.3)
```

```
## [1] 0.819538
```

```
###The proportion of total variance is 81.95%.
##d.
evec.3[,1:2]
```

```
##            [,1]        [,2]
## [1,] -0.3152136 -0.60269352
## [2,] -0.3248404 -0.47002261
## [3,] -0.3094965 -0.23076768
## [4,] -0.3120833 -0.05598404
## [5,] -0.3427922  0.07902824
## [6,] -0.4063244  0.29554455
## [7,] -0.4178300  0.29648919
## [8,] -0.3804563  0.42184529
```

```
###The first component is not interpretable, the second one explains 100m running result.
tra3.st<-matrix(0,ncol=8,nrow=55)
for (i in 1:55){
  for (j in 1:8){
    tra3.st[i,j]<-(tramps[i,j]-colMeans(tramps)[j])/sqrt(diag(S3)[j])
```

```
  }
}
z.3 <- tra3.st%*%evec.3[,1:2]
#plot(z.3[,1]~z.3[,2],xlab="PC1", ylab="PC2")

##e.
z1.3<-cbind(as.numeric(z.3[,1]),track[,1])
best5.3<-z1.3[order(z1.3[,1],decreasing = T),c(1,2)][1:5,]
worst5.3<-z1.3[order(z1.3[,1],decreasing = F),c(1,2)][1:5,]
###Best 5 are Cookis,Wsamoa,mauritiu,png,Singapor.(The slowest 5)
###Worst 5 are Usa,gbni,italy,ussr,gdr.(The fastest 5)

##f.
(cbind(best5.1,best5.2,best5.3))
```

```
##    as.numeric(z[, 1])   Country as.numeric(z.2[, 1]) Country
## 53           8.857144       usa             3.430556     usa
## 2            8.601844 australia             3.024230    gbni
## 30           8.113760     japan             2.726950   italy
## 44           8.066707  portugal             2.626851    ussr
## 38           7.833508  netherla             2.590092     gdr
##    as.numeric(z.3[, 1])  Country
## 53             9.646680   cookis
## 2              6.874246   wsamoa
## 30             4.370541 mauritiu
## 44             3.976622      png
## 38             3.272030 singapor
```

```
(cbind(worst5.1,worst5.2,worst5.3))
```

```
##    as.numeric(z[, 1])  Country as.numeric(z.2[, 1])  Country
## 12          -29.55186   cookis           -10.555626   cookis
## 55          -26.08673   wsamoa            -7.231216   wsamoa
## 46          -21.21484 singapor            -4.258658 mauritiu
## 16          -17.57318   domrep            -3.909193      png
## 35          -17.41341 malaysia            -3.122111 singapor
##    as.numeric(z.3[, 1]) Country
## 12            -3.609202     usa
## 55            -3.169869    gbni
## 46            -2.843512   italy
## 16            -2.733238    ussr
## 35            -2.698582     gdr
```

```
###Since problem 3 is using speed as a measuarement instead of time,
###the scores from problem 3 are opposite to previous problem 1 and 2.
###The top 5 in problem 2 are the same as the bottom 5 in problem 3, while
###The bottom 5 in problem 2 are the same as the top 5 in problem 3
###The least scores for 3 nations, Cookis, Wsamoa, Singapor are on the list for 3 problems.
###Especially, USA have highest scores problem 1,2 and lowest scores in problem 3.

#4.
##a.
#install.packages("MVN")
library(MVN)
```

```
## Warning: package 'MVN' was built under R version 3.4.4
```

```
## sROC 0.1-2 loaded
```

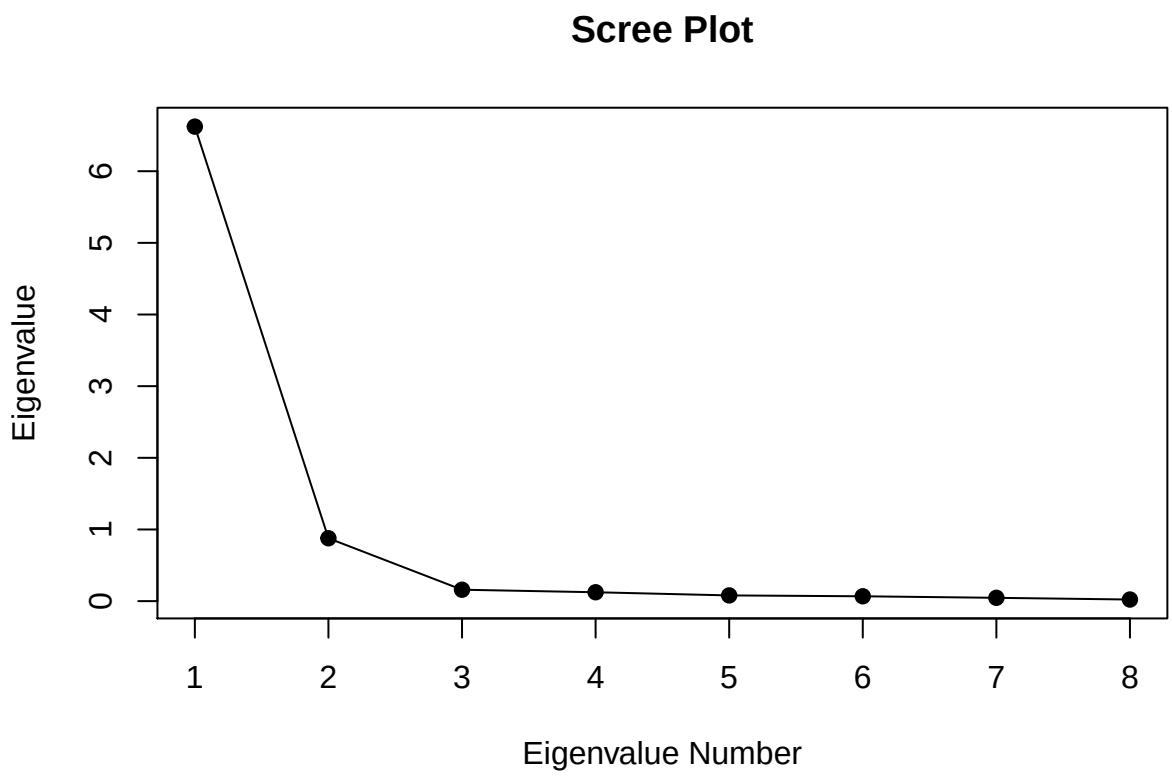
```
mvn(tra)
```

```
## $multivariateNormality
##              Test        Statistic              p value Result
## 1 Mardia Skewness 403.711887415238 2.18483318021084e-32     NO
## 2 Mardia Kurtosis 10.8691686325778                    0     NO
## 3             MVN             <NA>                 <NA>     NO
##
## $univariateNormality
##           Test  Variable Statistic   p value Normality
## 1 Shapiro-Wilk   100m       0.8273  <0.001      NO
## 2 Shapiro-Wilk   200m       0.9534  0.0327      NO
## 3 Shapiro-Wilk   400m       0.8890   1e-04      NO
## 4 Shapiro-Wilk   800m       0.8445  <0.001      NO
## 5 Shapiro-Wilk   1500m      0.8449  <0.001      NO
## 6 Shapiro-Wilk   5000m      0.8124  <0.001      NO
## 7 Shapiro-Wilk  10,00m      0.8031  <0.001      NO
## 8 Shapiro-Wilk Marathon     0.7877  <0.001      NO
##
## $Descriptives
##           n       Mean    Std.Dev Median    Min    Max    25th    75th
## 100m     55  10.471091 0.35142921  10.41   9.93  12.18  10.270  10.590
## 200m     55  20.940364 0.64464737  20.81  19.72  23.20  20.595  21.285
## 400m     55  46.438727 1.45701757  46.10  43.86  52.94  45.560  47.300
## 800m     55   1.793273 0.06368483   1.79   1.70   2.02   1.755   1.815
## 1500m    55   3.698182 0.15590941   3.64   3.51   4.24   3.600   3.770
## 5000m    55  13.845818 0.80116048  13.50  13.01  16.70  13.275  14.145
## 10,00m   55  28.989091 1.80773162  28.19  27.38  35.38  27.695  29.870
## Marathon 55 136.624000 9.22703347 132.35 128.22 164.70 130.705 139.300
##               Skew Kurtosis
## 100m     2.1995825 8.084650
## 200m     0.8629664 1.474384
## 400m     1.6101467 5.126249
## 800m     1.6743521 3.541918
## 1500m    1.6039970 3.011462
## 5000m    1.5387763 2.202239
## 10,00m   1.6001981 2.438012
## Marathon 1.4331296 1.042004
```

```
###Based on all variables reject the null hypothesis of Shapiro-Wilk Test
###It doesn't meet with the assumption of mulitivariate normality.
##b.
###It suggests to use PC methods.
##c.
R4<-cor(tra)
E <- eigen(R4)$vectors
Lambda <- diag(eigen(R4)$values)
diag(Lambda)
```

```
## [1] 6.62214613 0.87761829 0.15932114 0.12404939 0.07988027 0.06796515
## [7] 0.04641953 0.02260010
```

```
L <- E%*%sqrt(Lambda)
plot(1:8,diag(Lambda), xlab="Eigenvalue Number", ylab = "Eigenvalue",
     main= "Scree Plot", pch=19); lines(1:8, diag(Lambda))
```




```
percentage <- rep(0,p)
for (i in 1:8){
  percentage[i] <- sum(diag(Lambda)[1:i])/sum(diag(Lambda))
}
percentage
```

```
## [1] 0.8277683 0.9374706 0.9573857 0.9728919 0.9828769 0.9913725 0.9971750
## [8] 1.0000000
```

```
###1.The first eigenvalue explain 82.78%,
###2.Only 1 eigenvalue exceeds 1
###3.Scree plot shows m should be 1 or 2.

E1 <- E[,1]
Lambda1 <- Lambda[1,1]
L1 <- E1%*%t(sqrt(Lambda1))
C1 <- L1%*%t(L1)
Psi1 <- diag(diag(R4-C1))
(FA.PC1.res <-round(R4-(C1+Psi1),2))
```

```
##           100m  200m  400m  800m 1500m 5000m 10,00m Marathon
## 100m      0.00  0.21  0.09 -0.02 -0.08 -0.15  -0.14    -0.20
## 200m      0.21  0.00  0.06 -0.02 -0.06 -0.12  -0.12    -0.17
```

```
## 400m      0.09  0.06  0.00  0.00 -0.04 -0.08  -0.08    -0.10
## 800m     -0.02 -0.02  0.00  0.00  0.01 -0.03  -0.03    -0.03
## 1500m    -0.08 -0.06 -0.04  0.01  0.00  0.03   0.03     0.02
## 5000m    -0.15 -0.12 -0.08 -0.03  0.03  0.00   0.09     0.11
## 10,00m   -0.14 -0.12 -0.08 -0.03  0.03  0.09   0.00     0.11
## Marathon -0.20 -0.17 -0.10 -0.03  0.02  0.11   0.11     0.00
```

```
E2 <- E[,1:2]
Lambda2 <- Lambda[1:2,1:2]
L2 <- E2%*%sqrt(Lambda2)
C2 <- L2%*%t(L2)
Psi2 <- diag(diag(R4-C2))
(FA.PC2.res <-round(R4-(C2+Psi2),2))
```

```
##           100m  200m  400m  800m 1500m 5000m 10,00m Marathon
## 100m      0.00 -0.02 -0.03 -0.03 -0.01  0.01   0.01     0.02
## 200m     -0.02  0.00 -0.04 -0.02  0.00  0.01   0.00     0.01
## 400m     -0.03 -0.04  0.00  0.00 -0.01 -0.01  -0.01     0.00
## 800m     -0.03 -0.02  0.00  0.00  0.01 -0.02  -0.02    -0.02
## 1500m    -0.01  0.00 -0.01  0.01  0.00 -0.01  -0.01    -0.03
## 5000m     0.01  0.01 -0.01 -0.02 -0.01  0.00   0.01    -0.01
## 10,00m    0.01  0.00 -0.01 -0.02 -0.01  0.01   0.00    -0.01
## Marathon  0.02  0.01  0.00 -0.02 -0.03 -0.01  -0.01     0.00
```

```
library(psych)
```

```
## Warning: package 'psych' was built under R version 3.4.4
```

```
FA.PC1 <- principal(r=R4, nfactors=1, rotate="varimax")
diag(FA.PC1$residual)
```

```
##       100m       200m       400m       800m      1500m      5000m
## 0.33220873 0.24802223 0.16240680 0.09986492 0.07960654 0.12078756
##     10,00m   Marathon
## 0.10917488 0.22578222
```

```
FA.PC2 <- principal(r=R4, nfactors=2, rotate="varimax")
diag(FA.PC2$residual)
```

```
##       100m       200m       400m       800m      1500m      5000m
## 0.05018600 0.06100312 0.10831072 0.09972933 0.06245515 0.03534000
##     10,00m   Marathon
## 0.02653574 0.05667551
```

```
###By comparing m=1 and m=2, the residual matrix with m=2 has smaller residuals,
###We suggest to choose m=2.
##d.
FA.PC1$loadings
```

```
##
## Loadings:
##          PC1
## 100m     0.817
## 200m     0.867
## 400m     0.915
## 800m     0.949
## 1500m    0.959
## 5000m    0.938
```

```
## 10,00m   0.944
## Marathon 0.880
##
##                  PC1
## SS loadings    6.622
## Proportion Var 0.828
```

```
###Pc1 interpret all the running perfomances.
FA.PC2$loadings
```

```
##
## Loadings:
##          RC1   RC2
## 100m     0.277 0.934
## 200m     0.379 0.892
## 400m     0.545 0.771
## 800m     0.714 0.625
## 1500m    0.815 0.523
## 5000m    0.903 0.386
## 10,00m   0.905 0.394
## Marathon 0.936 0.258
##
##                  RC1   RC2
## SS loadings    4.202 3.298
## Proportion Var 0.525 0.412
## Cumulative Var 0.525 0.937
```

```
###In Pc2 the first pc interpret long run results(800m,1500m,5000m,10000m and Marathon)
###The second pc interpret short run results(100m,200m,400m,800m).
##e.
###Since the data are not normall distributed, we suggest to use Bartlett's method.
L1<-FA.PC2$loadings
D <- sqrt(solve(diag(diag(cov(tra)))))
ybar <- colMeans(tra)
(f <- solve(t(L1)%*%solve(Psi1)%*%L1)%*%t(L1)%*%solve(Psi1)%*%D%*%(tra[53,]-ybar))
```

```
##           [,1]
## RC1 -0.1742659
## RC2 -1.7993205
```

```
###For Usa, the score is PC1: -0.1743, PC2: -1.7993
###This shows that UsA has factors scores below 0, shows the perfomance of USA is below
###the average, that is, Usa sportsmen needed shorter time to finish the running race.
```