Jin Kweon - HW 4 (3032235207)

Jin Kweon

11/9/2017

Problem 1

Data import

```r
women <- read.delim("Data-HW4-track-women.dat", header = F, sep = "", na.strings = "")

men <- read.delim("Data-HW4-track-men.dat", header = F, sep = "", na.strings = "")

dim(men)
```

```
## [1] 54  9
```

```r
dim(women)
```

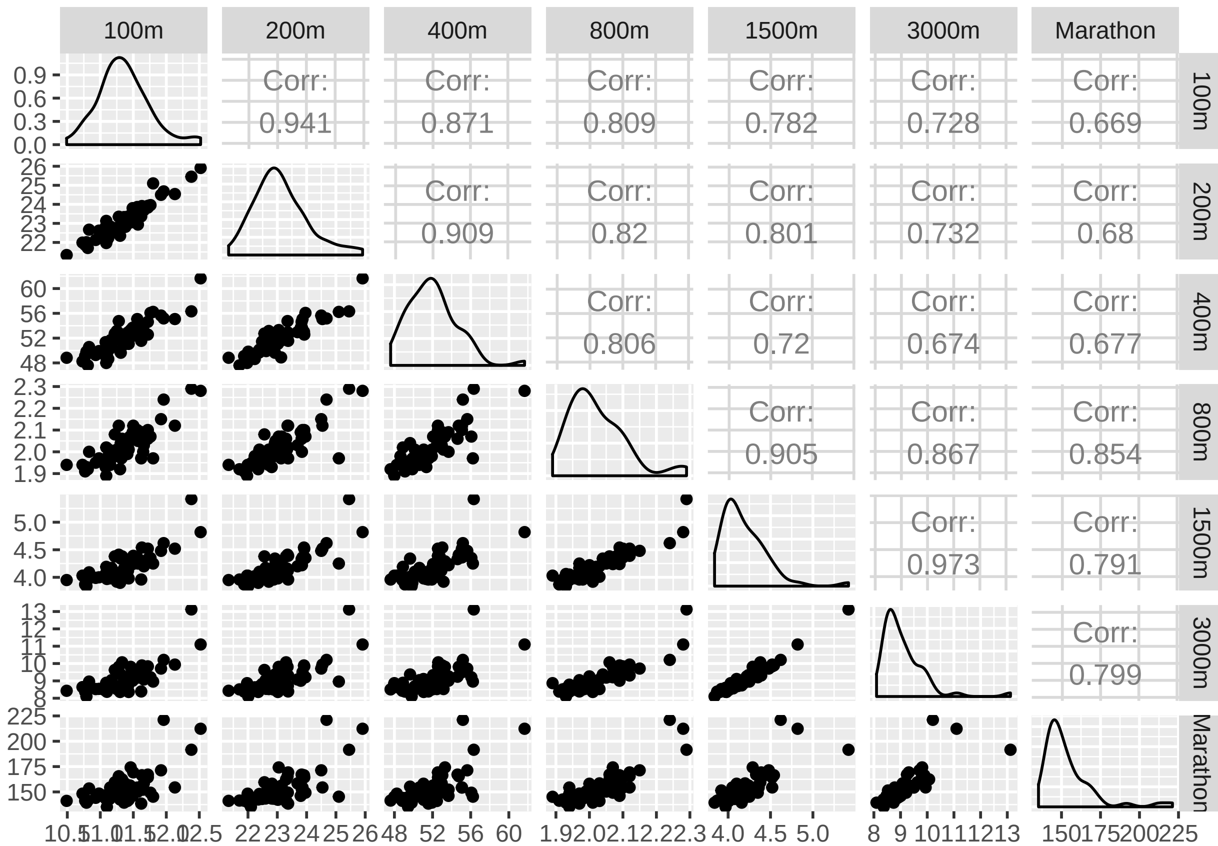
```
## [1] 54  8
```

```r
colnames(women) <- c("Country", "100m", "200m", "400m", "800m", "1500m", "3000m", "Marathon")
colnames(men) <- c("Country", "100m", "200m", "400m", "800m", "1500m", "5000m", "10000m", "Marathon")
```

Part a - Women

Obtain the sample correlation matrix for the women track records data, and determine its eigenvalues and eigenvectors.

```r
ggpairs(women[,-1])
```




```
cor(women[,-1])
```

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

```
scalewomen <- scale(women[,-1], T, T)
Rwomen <- cor(scalewomen)

loadingwomen <- eigen(Rwomen)$vectors
rownames(loadingwomen) <- colnames(scalewomen)
loadingwomen
```

```
##                [,1]       [,2]       [,3]        [,4]        [,5]
## 100m     -0.3777657 -0.4071756 -0.1405803  0.58706293 -0.16706891
```

```
## 200m     -0.3832103 -0.4136291 -0.1007833  0.19407501  0.09350016
## 400m     -0.3680361 -0.4593531  0.2370255 -0.64543118  0.32727328
## 800m     -0.3947810  0.1612459  0.1475424 -0.29520804 -0.81905467
## 1500m    -0.3892610  0.3090877 -0.4219855 -0.06669044  0.02613100
## 3000m    -0.3760945  0.4231899 -0.4060627 -0.08015699  0.35169796
## Marathon -0.3552031  0.3892153  0.7410610  0.32107640  0.24700821
##                 [,6]        [,7]
## 100m      0.53969730  0.08893934
## 200m     -0.74493139 -0.26565662
## 400m      0.24009405  0.12660435
## 800m     -0.01650651 -0.19521315
## 1500m    -0.18898771  0.73076817
## 3000m     0.24049968 -0.57150644
## Marathon -0.04826992  0.08208401
```

```r
eigenwomen <- eigen(Rwomen)$values
eigenwomen
```

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

```r
sum(eigenwomen)
```

```
## [1] 7
```

Comment:

The correlation matrix before and after the standardization should be the same!!!

The eigevenctors here are called "loadings." (it tells me the direction) It should be the same whatever ways I get upto the **sign difference.**

Eigenvalues are useful in determining proportion of variation. (it tells me the signicance of the direction)

The sum of eigenvalues are equal to the numberof columns. (it is 7 since the first column is just name of the countries)

Part b - Women

Determine the first two principal components for the standardized predictors. Find out the cumulative percentage of the total sample variance explained by the two components.

```r
#loadings...
loadingwomen[,1:2]
```

```
##                [,1]       [,2]
## 100m     -0.3777657 -0.4071756
## 200m     -0.3832103 -0.4136291
```

```
## 400m     -0.3680361 -0.4593531
## 800m     -0.3947810  0.1612459
## 1500m    -0.3892610  0.3090877
## 3000m    -0.3760945  0.4231899
## Marathon -0.3552031  0.3892153
```

```r
pcwomen <- scalewomen %*% loadingwomen
colnames(pcwomen) <- paste0("PC", 1:7)
rownames(pcwomen) <- women[,1]

head(pcwomen[,1:2])
```

```
##            PC1          PC2
## ARG -0.3932402 -0.131610654
## AUS  1.9316429  0.491067344
## AUT  1.2625204  0.193148352
## BEL  1.2917303 -0.002405316
## BER -1.3961086  0.760780551
## BRA  1.0067789  0.379516913
```

```r
#Check with prcomp
#head(prcomp(women[,-1], scale = T)$x[,1:2])

eigenwomen[1:2]
```

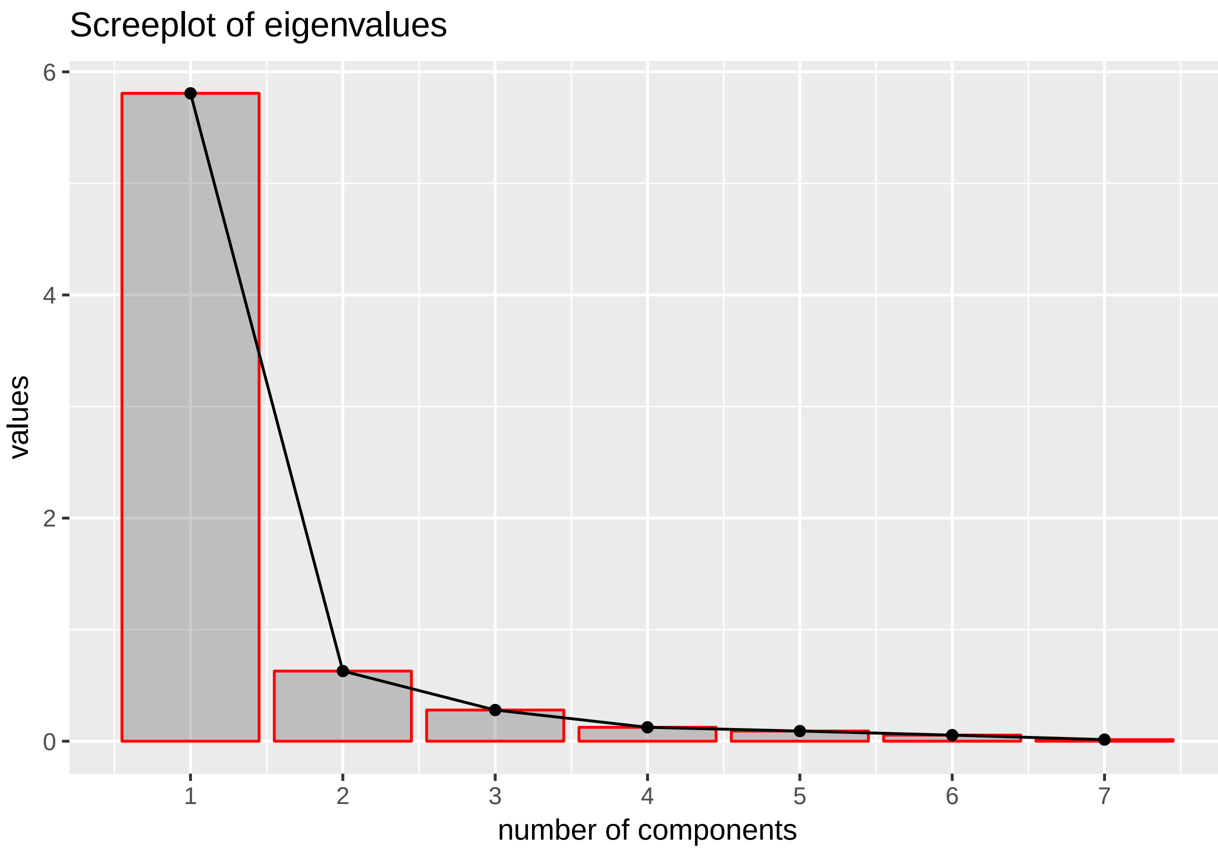
```
## [1] 5.8076245 0.6286934
```

```r
#Check with procomp
#as.vector((prcomp(women[,-1], scale = T)$sdev)^2)[1:2]

#Table of variance explained
eigen_data <- matrix(0, nrow = round(sum(eigenwomen),0), ncol = 3)
colnames(eigen_data) <- c("eigenvalue", "percentage", "cumulative.percentage")
rownames(eigen_data) <- paste0("comp", 1:sum(eigenwomen))

eigen_data[,1] <- eigenwomen
percentage <- apply(as.matrix(eigenwomen), 2, sum(eigenwomen), FUN = "/") * 100
eigen_data[,2] <- percentage

cum_fun <- function(x){ #x should be n * 1 column matrix
  for (i in 2:nrow(x)){
    x[i,] <- x[i-1,] + x[i,]
  }
  return(x)
}
cumulative <- cum_fun(percentage) #or use cumsum!!!
eigen_data[,3] <- cumulative

print(eigen_data)
```

```
##       eigenvalue percentage cumulative.percentage
## comp1 5.80762446 82.9660638              82.96606
## comp2 0.62869342  8.9813346              91.94740
## comp3 0.27933457  3.9904939              95.93789
## comp4 0.12455472  1.7793531              97.71725
## comp5 0.09097174  1.2995963              99.01684
```

```
## comp6 0.05451882  0.7788403              99.79568
## comp7 0.01430226  0.2043181             100.00000
```

```
graph <- ggplot(as.data.frame(eigen_data[,1]), aes(x = 1:7, y = as.numeric(eigen_data[,1])))
graph <- graph + geom_bar(stat = "identity", alpha = 0.3, color = "red") + geom_point() +
  geom_line() +
  labs(title = "Screeplot of eigenvalues", x = "number of components", y = "values") +
  scale_x_continuous(breaks=seq(1,12,1))
graph
```



Comment:

Again Z should be the same no matter which way I used, upto the sign difference. Please check my output above for my first two PCs.

Again, we need to use the formula $\frac{\lambda_i}{\lambda_1 + \ldots + \lambda_p}$ is the proportion of variance captured by i-th principal components, when i = 1, . . . , p.

The cumulative percentage of the total sample variance explained by the two components is around 91.95 %.

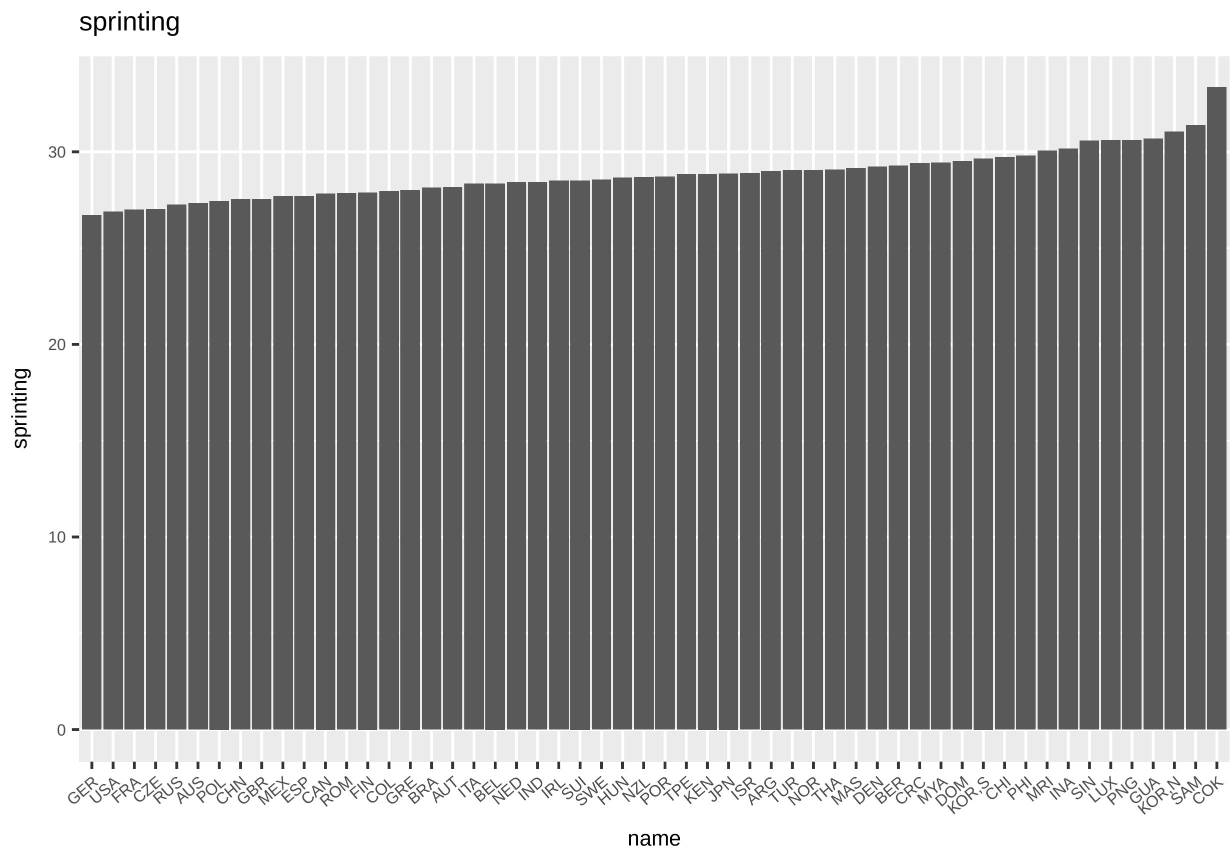
I also made a scree-plot, which is one of the ways to choose how many PCs I should use. (Personally, I am not a fan of scree-plot, since it is too subjective. I prefer predetermined amount of variation, Kaiser's rule, or Jolife's rule.)

I think I will only need two dimensions of PCs for this data.

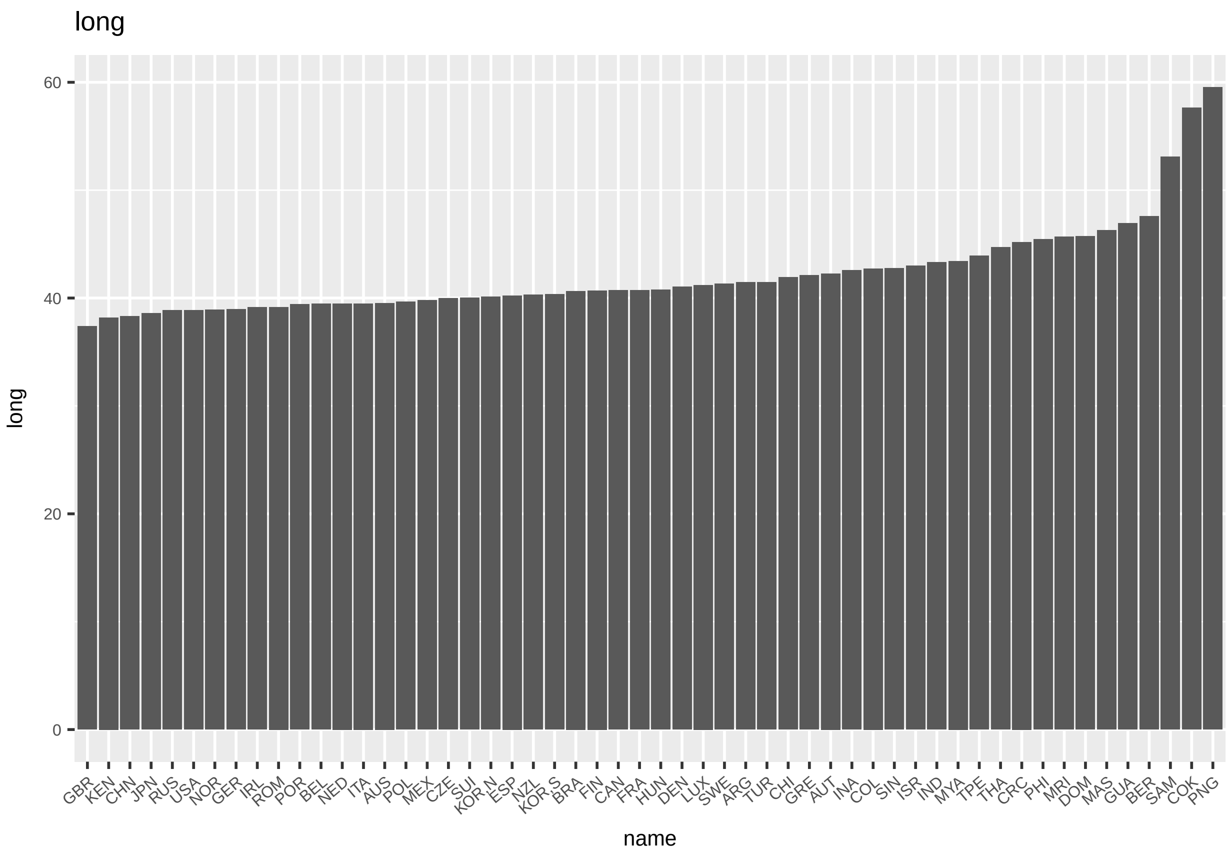
Part c - Women

Interpret the two principal components (and loadings).

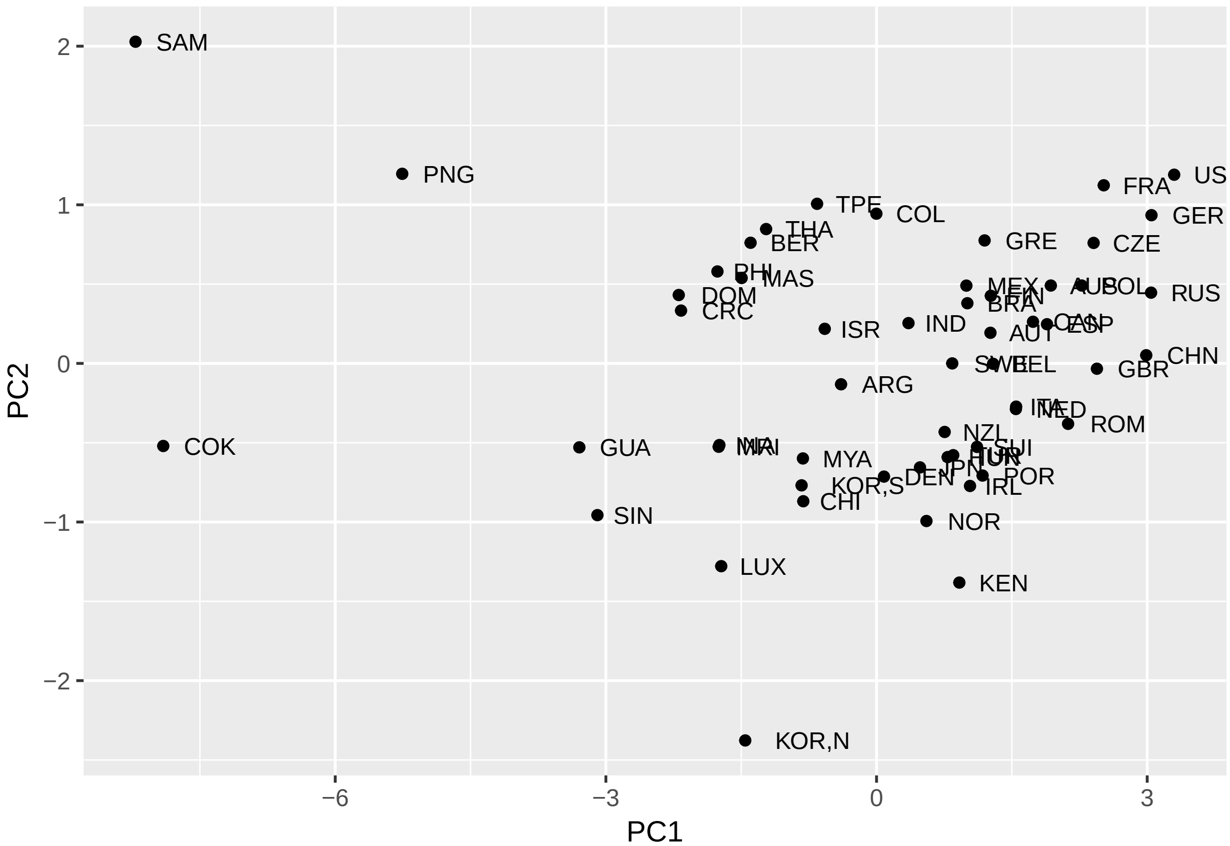
```
women$sprinting <- apply(women[,2:4], 1, mean)
women$long <- apply(women[,5:8], 1, mean)

womensprint <- data.frame(sprinting = women[,9])
womenlong <- data.frame(long = women[,10])

womensprint$Rank <- rank(womensprint$sprinting)
rownames(womensprint) <- women[,1]
womenlong$Rank <- rank(womenlong$long)
rownames(womenlong) <- women[,1]

womensprintorder <- womensprint[order(womensprint$Rank), ]
womensprintorder$name <- rownames(womensprintorder)
womensprintorder$name <- factor(womensprintorder$name,
                                levels = womensprintorder$name[order(womensprintorder$sprinting)])


womenlongorder <- womenlong[order(womenlong$Rank), ]
womenlongorder$name <- rownames(womenlongorder)
womenlongorder$name <- factor(womenlongorder$name,
                              levels = womenlongorder$name[order(womenlongorder$long)])


ggplot(womensprintorder, aes(x = name, y = sprinting)) + geom_bar(stat = "identity") +
  theme(text = element_text(size=8),axis.text.x = element_text(angle = 40, hjust = 1)) +
  ggtitle("sprinting")
```




```
ggplot(womenlongorder, aes(x = name, y = long)) + geom_bar(stat = "identity") +
  theme(text = element_text(size=8), axis.text.x = element_text(angle = 40, hjust = 1)) +
  ggtitle("long")
```

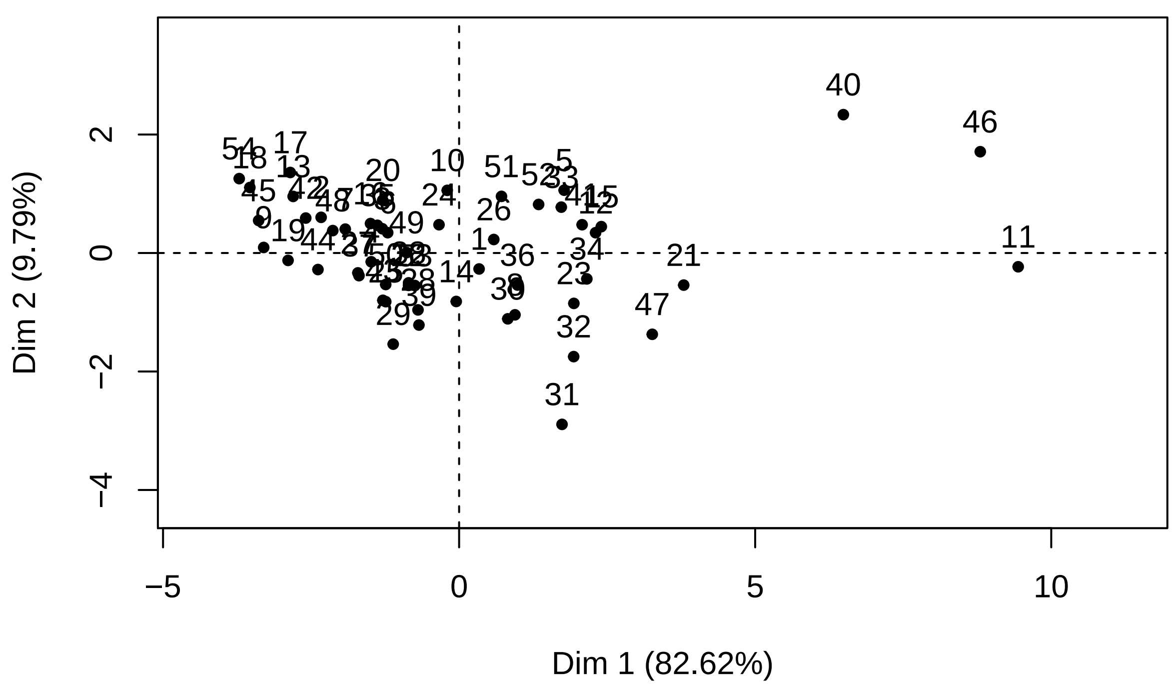



```r
womenpca <- prcomp(women[,-1], scale = T)
# fviz_pca_ind(womenpca,
#              col.ind = "cos2", # Color by the quality of representation
#              gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
#              repel = TRUE     # Avoid text overlapping
#              )
ggplot(as.data.frame(pcwomen[,1:2]), aes(x = PC1, y = PC2)) + geom_point() +
  geom_text(aes(label = rownames(pcwomen), hjust = -0.4), size = 3)
```

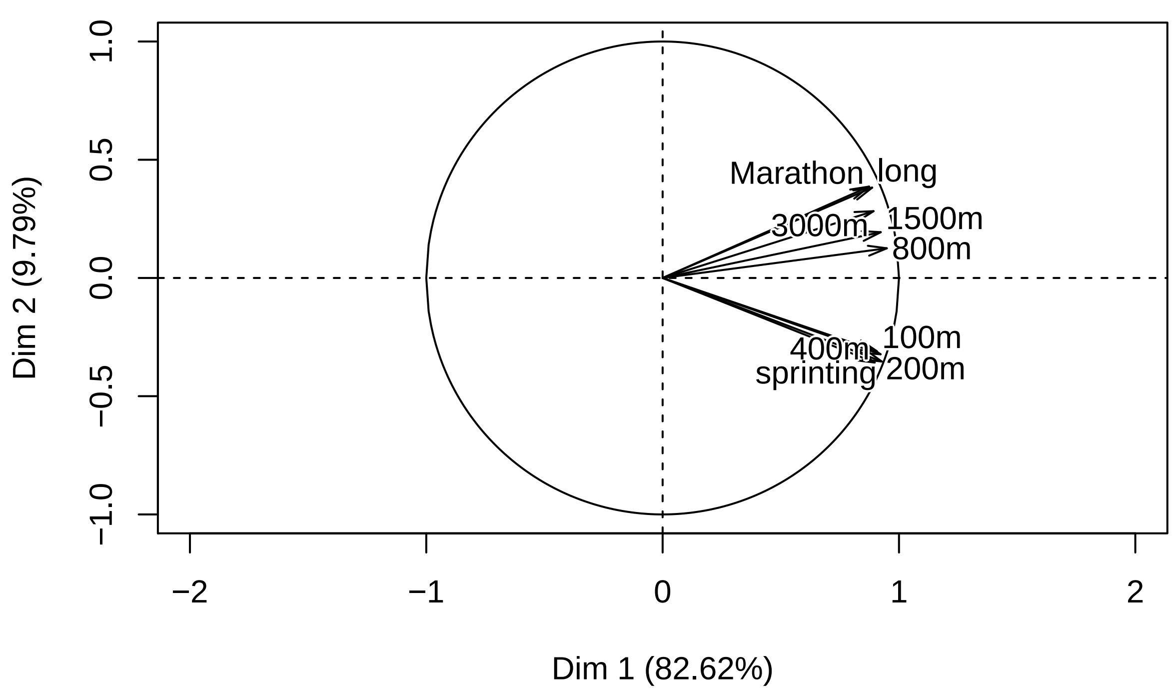


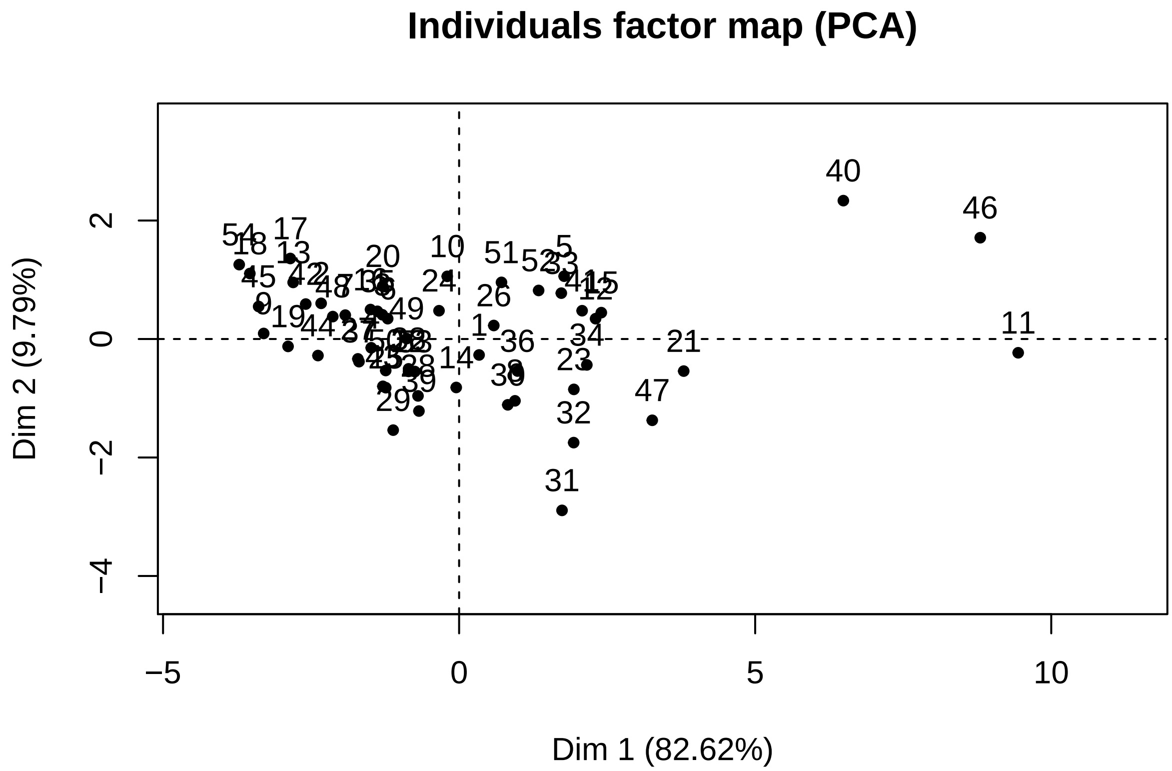

```r
plot(PCA(women[,-1]))
```

Individuals factor map (PCA)



Variables factor map (PCA)





Comment:

Please also refer to the graph I made for part d).

First of all, I think by looking at the scree-plot, I feel like I only need two PCs to explain the data well enough (looking for elbow).

Second, when I see the graphs, most of the countries are left-skwed… And, as USA, FRA, and GER (the countries who have good amounts of athletes) are placed in the right-side, I can tell that the countries who have done pretty well in runnings are placed in the right-side of PC1. Only three countries: SAM, COK, PNG are on the left-hand side.

I think the first component indicates how good each coungry is in the short distance (sprinting) like for example in 100m, 200m, and 400m. (I ranked the long distances and short distances.) It is quite clear as USA, GER, FRA, and RUS aer all placed in the top tiers; however, for example, KEN is not really having big number in PC1 as they are doing well in the long distances but no in the short distances. (However, in overall, **PC1 shows atheletic excellence.**)

The second component is not clear to be interpreted, and this makes sense as this component does not take small variance, as we saw in the previous question (eigenvalue). However, one thing I found that might be possible interpretation for PC2 shows how big the **difference is betwene short and long distances running (so how countries are good at long run compared to short run)**… For example, KEN and KOR.N show the good amounts of gaps between short and long runnings, but USA and RUS did not… However, this is not a perfect interpretation…

PC1: **shows atheletic excellence.**

PC2: **difference is betwene short and long distances running (so how countries are good at long run compared to short run)**

Part d - Women

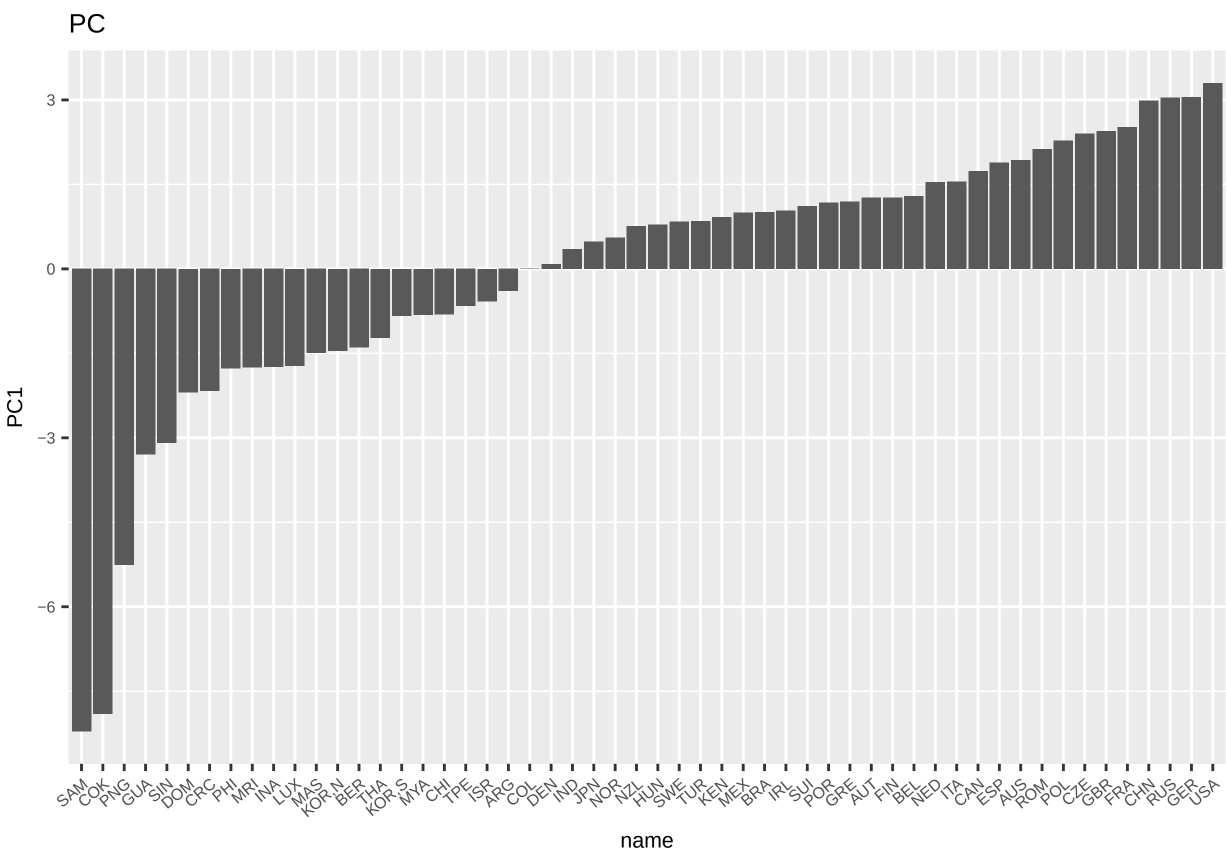
Rank the nations based on their score on the first principal component. Does this ranking correspond with your intuitive notion of athletic excellence for the various countries?

```
pcwomenrank <- data.frame(PC1 = pcwomen[,1])
pcwomenrank$Rank <- rank(pcwomenrank$PC1)
order <- pcwomenrank[order(pcwomenrank$Rank), ]
order$name <- rownames(order)
order$name <- factor(order$name, levels = order$name[order(order$PC1)])
order
```

```
##                PC1 Rank  name
## SAM   -8.213415123    1   SAM
## COK   -7.906227224    2   COK
## PNG   -5.257449747    3   PNG
## GUA   -3.294123799    4   GUA
## SIN   -3.093919517    5   SIN
## DOM   -2.192409809    6   DOM
## CRC   -2.166811506    7   CRC
## PHI   -1.763533682    8   PHI
## MRI   -1.749727754    9   MRI
## INA   -1.741942057   10   INA
## LUX   -1.721467731   11   LUX
## MAS   -1.495210140   12   MAS
## KOR,N -1.455347346   13 KOR,N
## BER   -1.396108552   14   BER
## THA   -1.223805050   15   THA
## KOR,S -0.830794629   16 KOR,S
## MYA   -0.815981458   17   MYA
## CHI   -0.811838204   18   CHI
## TPE   -0.659093139   19   TPE
## ISR   -0.574161730   20   ISR
## ARG   -0.393240234   21   ARG
## COL   -0.001927672   22   COL
## DEN    0.082495533   23   DEN
## IND    0.354256642   24   IND
## JPN    0.481657610   25   JPN
## NOR    0.553003461   26   NOR
## NZL    0.755235487   27   NZL
## HUN    0.788251063   28   HUN
## SWE    0.839149567   29   SWE
## TUR    0.850127798   30   TUR
## KEN    0.917735409   31   KEN
## MEX    0.995766285   32   MEX
## BRA    1.006778878   33   BRA
```

```
## IRL    1.035907216   34   IRL
## SUI    1.113545239   35   SUI
## POR    1.175249957   36   POR
## GRE    1.197800425   37   GRE
## AUT    1.262520373   38   AUT
## FIN    1.266731340   39   FIN
## BEL    1.291730279   40   BEL
## NED    1.544760622   41   NED
## ITA    1.547452839   42   ITA
## CAN    1.734340591   43   CAN
## ESP    1.889462264   44   ESP
## AUS    1.931642887   45   AUS
## ROM    2.123005711   46   ROM
## POL    2.273765780   47   POL
## CZE    2.406030321   48   CZE
## GBR    2.442706280   49   GBR
## FRA    2.518345696   50   FRA
## CHN    2.989466907   51   CHN
## RUS    3.042948214   52   RUS
## GER    3.047516603   53   GER
## USA    3.299148823   54   USA
```

```
ggplot(order, aes(x = name, y = PC1)) + geom_bar(stat = "identity") +
  theme(text = element_text(size=8), axis.text.x = element_text(angle = 40, hjust = 1)) +
  ggtitle("PC")
```



Comment:

As I commented in the previous question, I think this ranking shows the notion of athletic excellence in the **sprinting.** Most of the countires who have high PC here have good records in short distance runnings. For example, although KEN is one of the countries who have shown the excellence in marathon does not perform high in PC1. (However, as I also mentioned in the previous question, PC1 shows atheletic excellence for the various countries pretty well) USA, GER, RUS, CHN, GRB, and FRA are on the top rankings.

Part e - Women

Convert the national track records for women to speeds measured in meters per second. Perform a principal components analysis using the covariance matrix of the speed data. Compare the results with the results in (b). Do your interpretations of the components differ? If the nations are ranked on the basis of their score on the first principal component, does the subsequent ranking differ from that in (d)? Which analysis do you prefer? Why?

```
women1 <- 100 / women[,2]
women2 <- 200 / women[,3]
women3 <- 400 / women[,4]
women4 <- 800 / (women[,5] * 60)
women5 <- 1500 / (women[,6] * 60)
women6 <- 3000 / (women[,7] * 60)
women7 <- 42195 / (women[,8] * 60)
womene <- data.frame(`100m` = women1, `200m` = women2, `400m` = women3, `800m` = women4,
                     `1500m` = women5, `3000m` = women6, marathon = women7)



#A
cov(womene)
```

```
##               X100m      X200m      X400m      X800m     X1500m     X3000m
## X100m    0.09053826 0.09560635 0.09667244 0.06506402 0.08221980 0.09214221
## X200m    0.09560635 0.11467144 0.11386990 0.07492487 0.09601895 0.10543645
## X400m    0.09667244 0.11386990 0.13778886 0.08094090 0.09544299 0.10831645
## X800m    0.06506402 0.07492487 0.08094090 0.07352284 0.08645423 0.09975466
## X1500m   0.08221980 0.09601895 0.09544299 0.08645423 0.12384050 0.14371481
## X3000m   0.09214221 0.10543645 0.10831645 0.09975466 0.14371481 0.17658433
## marathon 0.08109987 0.09331033 0.10188073 0.09430563 0.11845777 0.14656043
##            marathon
## X100m    0.08109987
## X200m    0.09331033
## X400m    0.10188073
## X800m    0.09430563
## X1500m   0.11845777
## X3000m   0.14656043
## marathon 0.16671409
```

```r
covwomen <- cov(scale(womene, T, F)) #should be the same!!!

loadingwomene <- eigen(covwomen)$vectors
rownames(loadingwomene) <- colnames(womene)
-1 * loadingwomene # I multipled by -1 just to make the first column to be positive...
```

```
##                [,1]        [,2]        [,3]        [,4]        [,5]
## X100m    0.3102442  0.37596510  0.09755628 -0.58479630 -0.04613051
## X200m    0.3573948  0.43376925  0.08896099 -0.32287531 -0.02977941
## X400m    0.3787367  0.51873227 -0.27424547  0.66667306 -0.18727340
## X800m    0.2993405 -0.05313551 -0.05252266  0.12808676  0.89434367
## X1500m   0.3912131 -0.21084397  0.43498609  0.05510789  0.12725405
## X3000m   0.4595909 -0.39557338  0.42664455  0.18388862 -0.35674301
## marathon 0.4227291 -0.44458346 -0.73031571 -0.23675670 -0.13639673
##                 [,6]        [,7]
## X100m    -0.62433141 -0.13775753
## X200m     0.68870961  0.31103524
## X400m    -0.12377209 -0.13198849
## X800m    -0.13592439  0.26472817
## X1500m    0.23626094 -0.73364469
## X3000m   -0.19925854  0.49948755
## marathon  0.08106294 -0.09516116
```

```r
eigenwomene <- eigen(covwomen)$values
eigenwomene
```

```
## [1] 0.732146965 0.086071850 0.033380034 0.014977343 0.008851016 0.006167575
## [7] 0.002065542
```

```r
#B
-1 * loadingwomene[,1:2]
```

```
##                [,1]        [,2]
## X100m    0.3102442  0.37596510
## X200m    0.3573948  0.43376925
## X400m    0.3787367  0.51873227
## X800m    0.2993405 -0.05313551
## X1500m   0.3912131 -0.21084397
## X3000m   0.4595909 -0.39557338
## marathon 0.4227291 -0.44458346
```

```r
pcwomene <- scale(womene, T, F) %*% loadingwomene
colnames(pcwomene) <- paste0("PC", 1:7)
rownames(pcwomene) <- women[,1]

head(pcwomene[,1:2])
```

```
##            PC1         PC2
## ARG  0.1635073  0.04692099
## AUS -0.7307601 -0.19835239
## AUT -0.3667764 -0.13521031
## BEL -0.4429985  0.02515002
## BER  0.6651627 -0.34274202
## BRA -0.2903061 -0.15852299
```

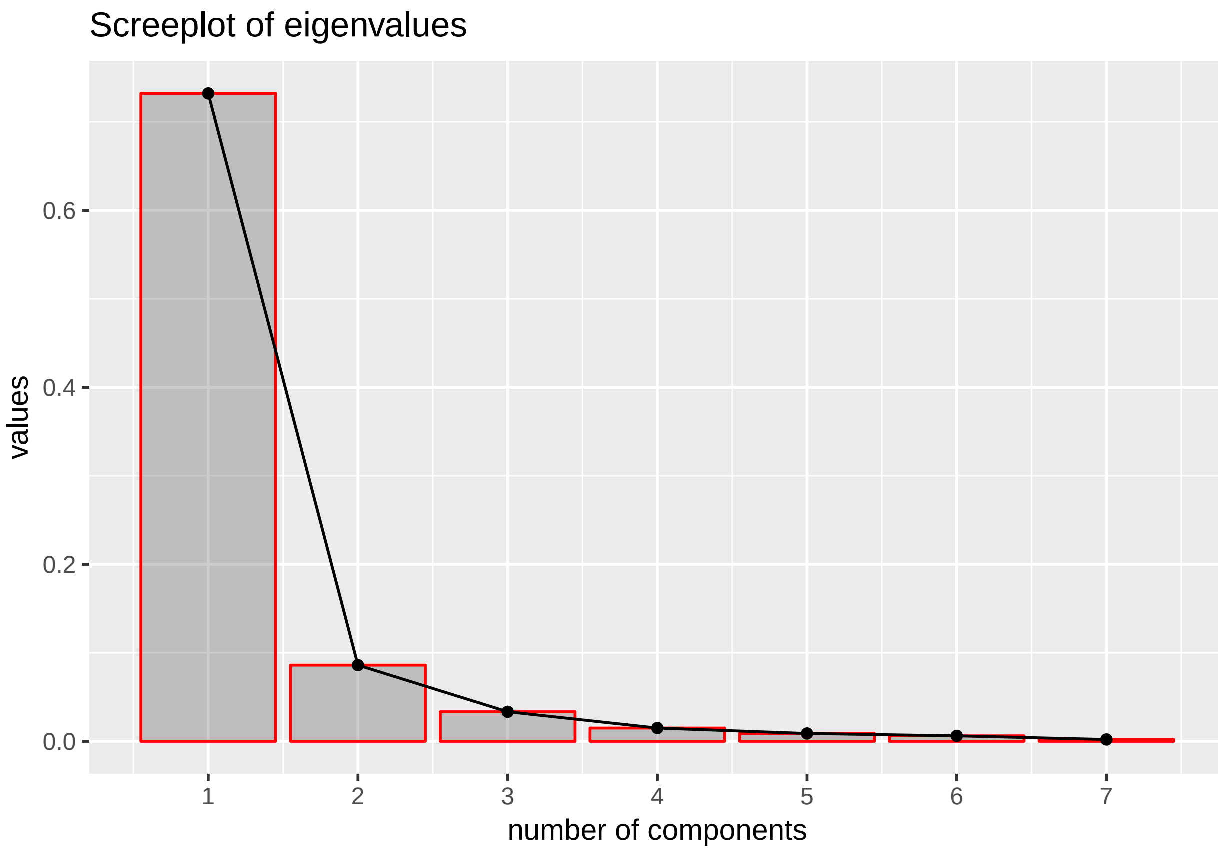
```r
#Check with prcomp
#head(prcomp(womene, center = T)$x[,1:2])

eigenwomene[1:2]
```

```
## [1] 0.73214696 0.08607185
```

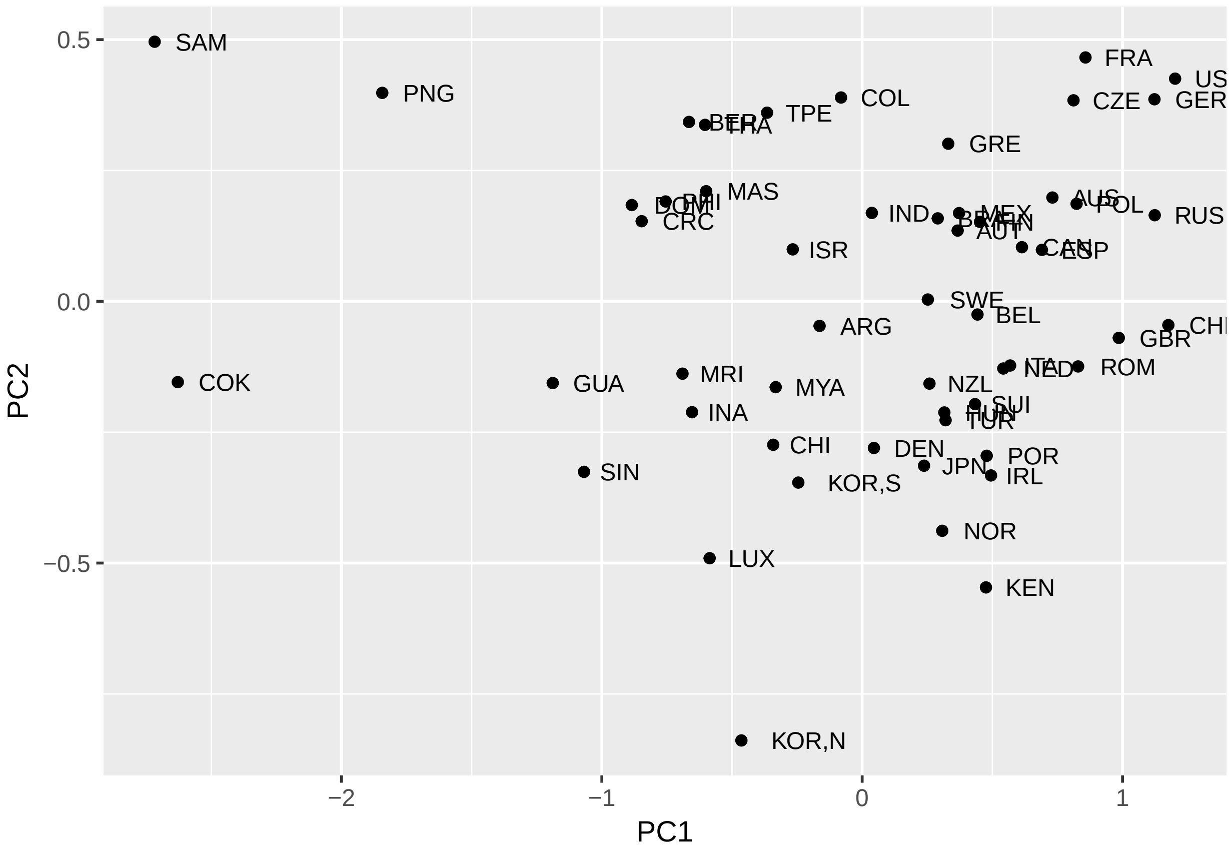
```r
#Check with procomp
#as.vector((prcomp(womene, center = T)$sdev)^2)[1:2]

#Table of variance explained
eigen_data <- matrix(0, nrow = 7, ncol = 3)
colnames(eigen_data) <- c("eigenvalue", "percentage", "cumulative.percentage")
rownames(eigen_data) <- paste0("comp", 1:7)

eigen_data[,1] <- eigenwomene
percentage <- apply(as.matrix(eigenwomene), 2, sum(eigenwomene), FUN = "/") * 100
eigen_data[,2] <- percentage

cum_fun <- function(x){ #x should be n * 1 column matrix
  for (i in 2:nrow(x)){
    x[i,] <- x[i-1,] + x[i,]
  }
  return(x)
}
cumulative <- cum_fun(percentage) #or use cumsum!!!
eigen_data[,3] <- cumulative

print(eigen_data)
```

```
##        eigenvalue percentage cumulative.percentage
## comp1 0.732146965 82.8538913              82.85389
## comp2 0.086071850  9.7403774              92.59427
## comp3 0.033380034  3.7774734              96.37174
## comp4 0.014977343  1.6949208              98.06666
## comp5 0.008851016  1.0016310              99.06829
## comp6 0.006167575  0.6979577              99.76625
## comp7 0.002065542  0.2337484             100.00000
```

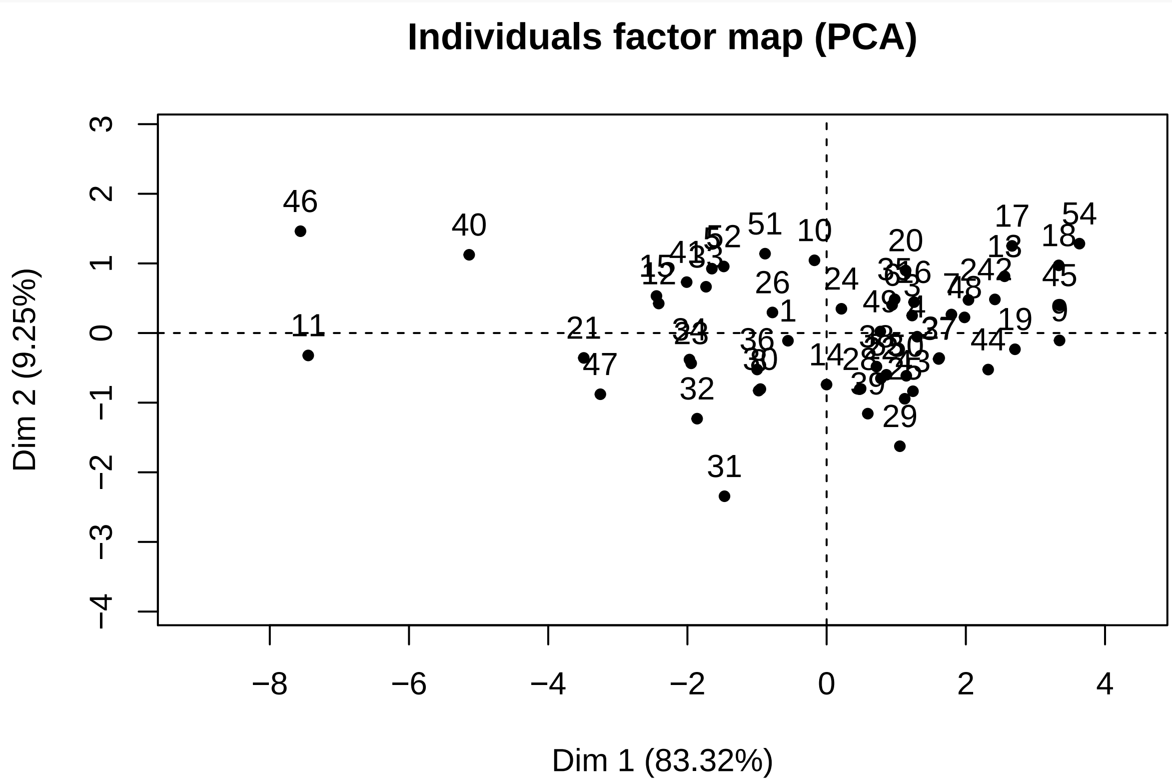
```r
graph <- ggplot(as.data.frame(eigen_data[,1]), aes(x = 1:7, y = as.numeric(eigen_data[,1])))
graph <- graph + geom_bar(stat = "identity", alpha = 0.3, color = "red") + geom_point() +
  geom_line() +
  labs(title = "Screeplot of eigenvalues", x = "number of components", y = "values") +
  scale_x_continuous(breaks=seq(1,12,1))
graph
```

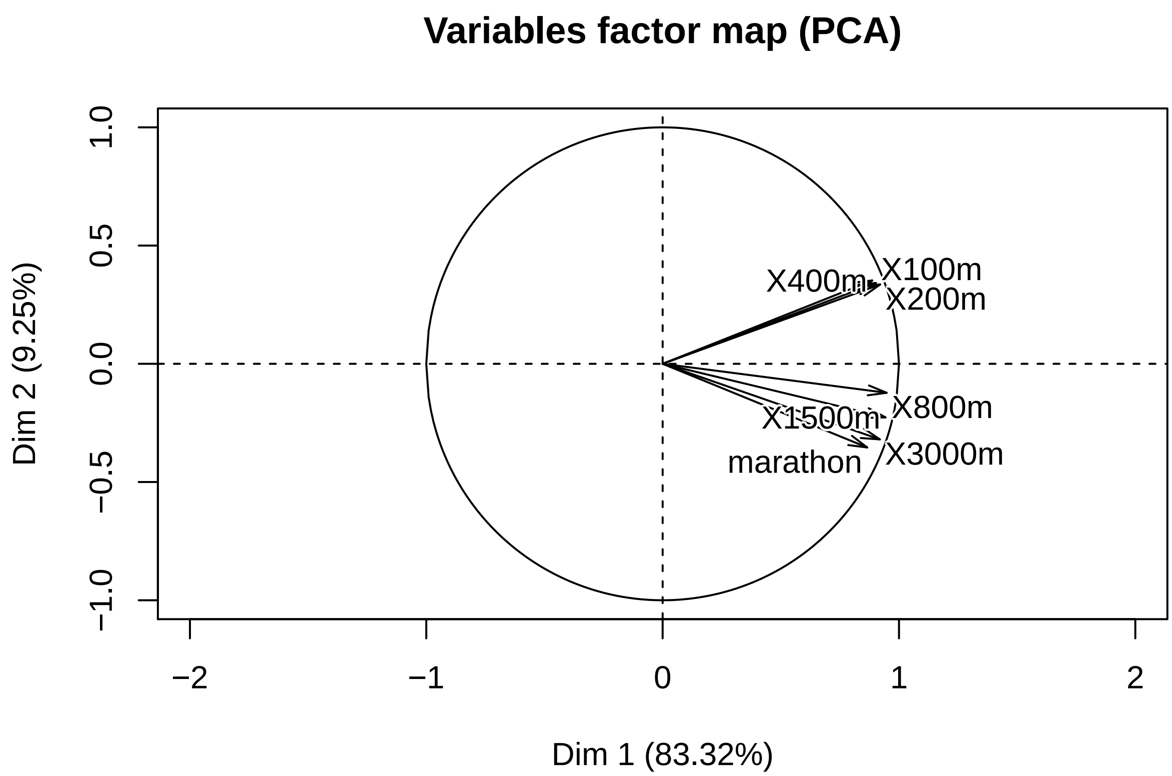


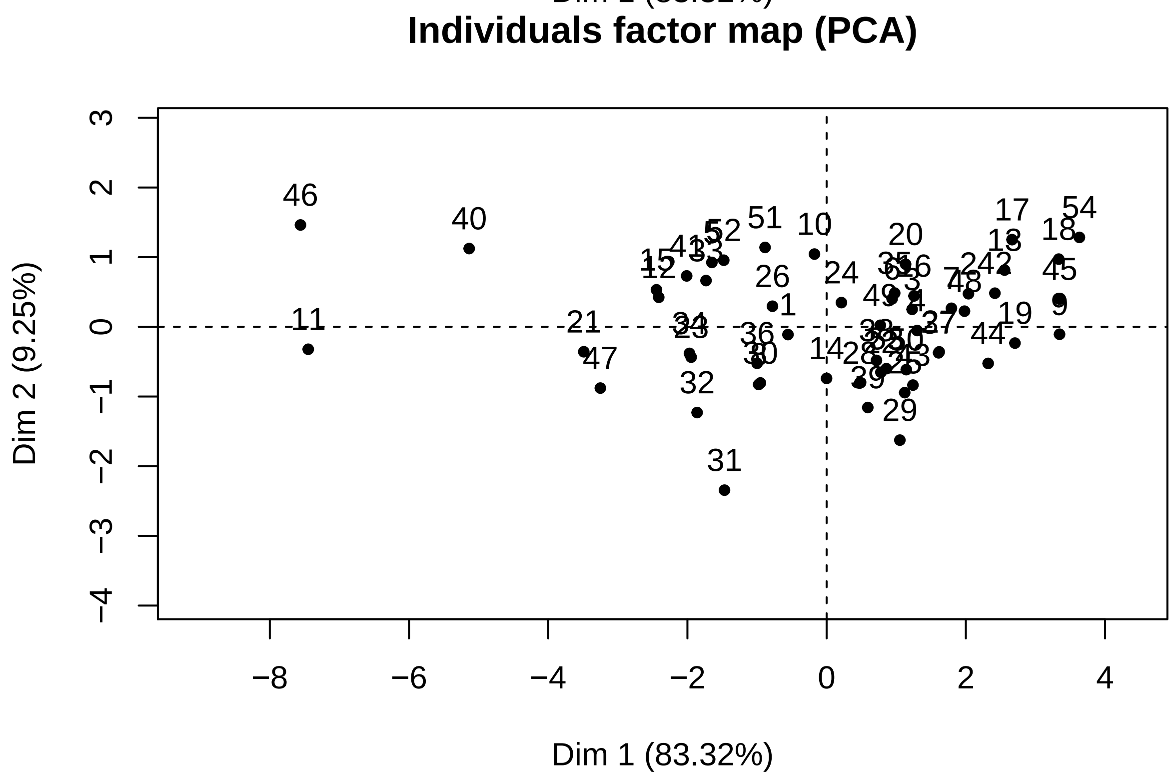

```
#C
womenpcae <- prcomp(womene, center = T)
# fviz_pca_ind(womenpcae,
#              col.ind = "cos2", # Color by the quality of representation
#              gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
#              repel = TRUE     # Avoid text overlapping
#              )
ggplot(as.data.frame(-1 *pcwomene[,1:2]), aes(x = PC1, y = PC2)) + geom_point() +
  geom_text(aes(label = rownames(pcwomene), hjust = -0.4), size = 3)
```



```r
plot(PCA(womene))
```







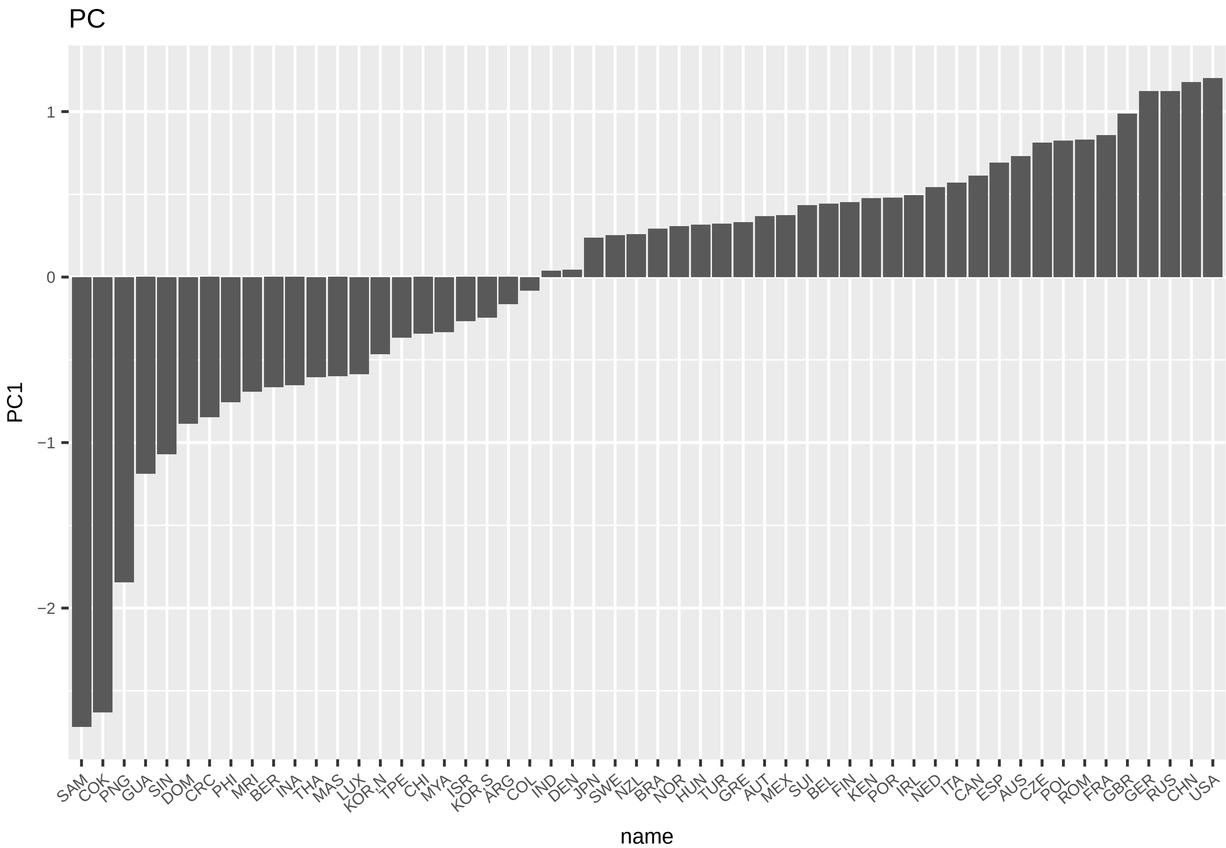

```r
#D
pcwomenranke <- data.frame(PC1 = -1 * pcwomene[,1])
pcwomenranke$Rank <- rank(pcwomenranke$PC1)
order <- pcwomenranke[order(pcwomenranke$Rank), ]
order$name <- rownames(order)
```

```r
order$name <- factor(order$name, levels = order$name[order(order$PC1)])
order
```

```
##               PC1 Rank  name
## SAM   -2.71771807    1   SAM
## COK   -2.62879675    2   COK
## PNG   -1.84336993    3   PNG
## GUA   -1.18861811    4   GUA
## SIN   -1.06842518    5   SIN
## DOM   -0.88504633    6   DOM
## CRC   -0.84692910    7   CRC
## PHI   -0.75485826    8   PHI
## MRI   -0.69018886    9   MRI
## BER   -0.66516272   10   BER
## INA   -0.65339346   11   INA
## THA   -0.60359791   12   THA
## MAS   -0.59922206   13   MAS
## LUX   -0.58590351   14   LUX
## KOR,N -0.46389493   15 KOR,N
## TPE   -0.36531542   16   TPE
## CHI   -0.34166630   17   CHI
## MYA   -0.33216347   18   MYA
## ISR   -0.26658096   19   ISR
## KOR,S -0.24536986   20 KOR,S
## ARG   -0.16350734   21   ARG
## COL   -0.08116611   22   COL
## IND    0.03711235   23   IND
## DEN    0.04509413   24   DEN
## JPN    0.23769282   25   JPN
## SWE    0.25226734   26   SWE
## NZL    0.25856583   27   NZL
## BRA    0.29030612   28   BRA
## NOR    0.30759786   29   NOR
## HUN    0.31568566   30   HUN
## TUR    0.32048183   31   TUR
## GRE    0.33072156   32   GRE
## AUT    0.36677641   33   AUT
## MEX    0.37221554   34   MEX
## SUI    0.43345412   35   SUI
## BEL    0.44299854   36   BEL
## FIN    0.45266993   37   FIN
## KEN    0.47549223   38   KEN
## POR    0.47833489   39   POR
## IRL    0.49491174   40   IRL
## NED    0.54210775   41   NED
## ITA    0.56839218   42   ITA
## CAN    0.61373983   43   CAN
## ESP    0.69043601   44   ESP
## AUS    0.73076013   45   AUS
## CZE    0.81183964   46   CZE
## POL    0.82359316   47   POL
## ROM    0.82951735   48   ROM
## FRA    0.85773396   49   FRA
## GBR    0.98571205   50   GBR
```

```
## GER     1.12276551   51   GER
## RUS     1.12377157   52   RUS
## CHN     1.17615030   53   CHN
## USA     1.20199632   54   USA
```

```r
ggplot(order, aes(x = name, y = PC1)) + geom_bar(stat = "identity") +
  theme(text = element_text(size=8), axis.text.x = element_text(angle = 40, hjust = 1)) +
  ggtitle("PC")
```



Comment:

As professor recomended, I will use mean-centered (not standardized) for covariance matrix, and standardized matrix for correlation matrix.

First of all, I want to mention I adjusted the sign, for interpretation and visual purpose. In PCA, interpretation and visual are both really important, so I was extra careful about them. . .

Definitely, as I used the covaraince matrix, the eigenvalues are different;however, the cumulative percentages are almost the same/similar. And, the two principal components are different for sure.

As you can easily see from my bar plots, the nations' ranks based on the scores on the first principal component, the bar plot looks almost the same! Thus, the rankings are not significantly different.

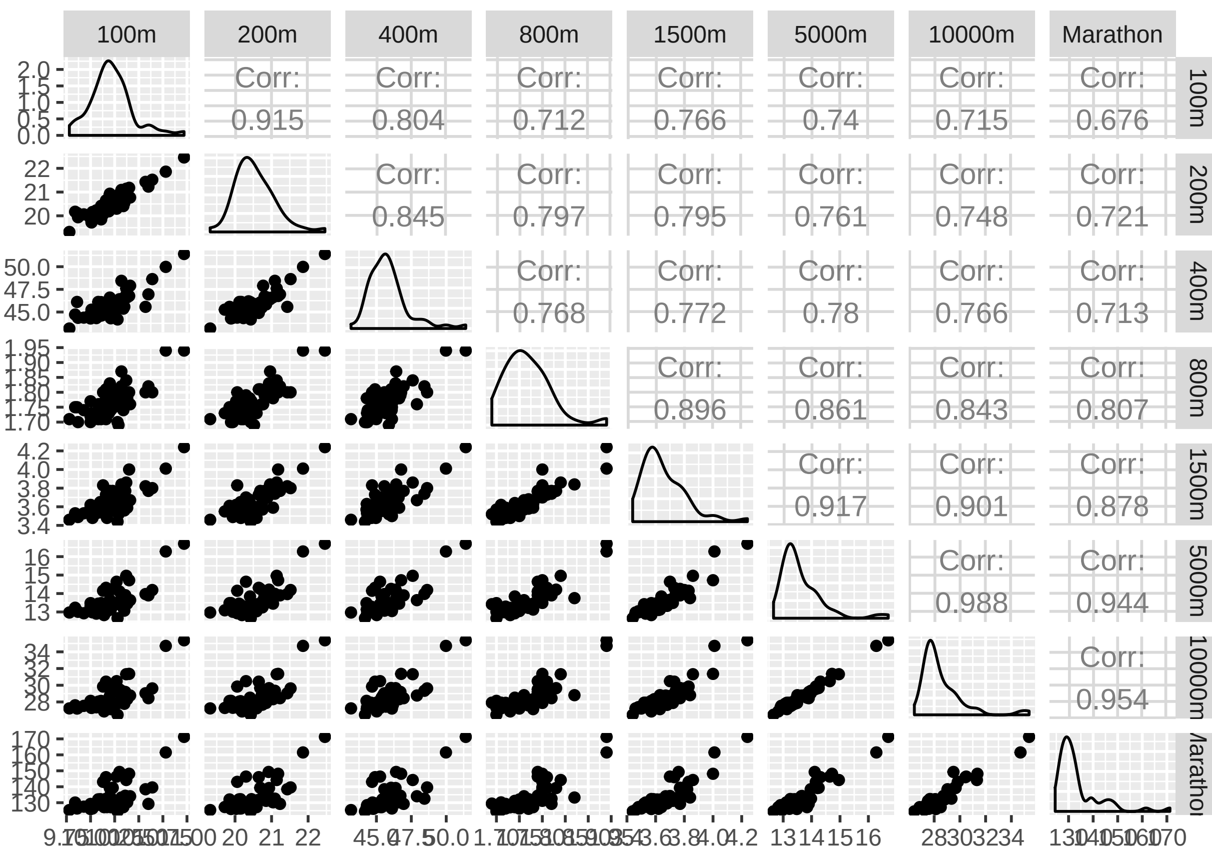
Furthermore, the interpretation of the components are also the same. It can be easily found on the PC1 v.s. PC2 plots.

I personally do not have any preference. The difference is basically the "unit" and "correlation matrix or covariance matrix." However, both of them can be used in different ways, and both are useful. For example, I personally prefer to use correlation matrix most of the times when I want to know the total sample variance explained by the component, as the sum of all eigen-values are added upto the number of variables. And, also this might be easier to interpret it, since correlation is always between 0 and 1. However, e) is also changing "unit," so I can do better analysis when I try to compare speeds between speed of each distance.

Part f

Part a - Men

```r
ggpairs(men[,-1])
```



```r
cor(men[,-1])
```

```
##              100m      200m      400m      800m     1500m     5000m
## 100m    1.0000000 0.9147554 0.8041147 0.7119388 0.7657919 0.7398803
```

```
## 200m     0.9147554 1.0000000 0.8449159 0.7969162 0.7950871 0.7613028
## 400m     0.8041147 0.8449159 1.0000000 0.7677488 0.7715522 0.7796929
## 800m     0.7119388 0.7969162 0.7677488 1.0000000 0.8957609 0.8606959
## 1500m    0.7657919 0.7950871 0.7715522 0.8957609 1.0000000 0.9165224
## 5000m    0.7398803 0.7613028 0.7796929 0.8606959 0.9165224 1.0000000
## 10000m   0.7147921 0.7479519 0.7657481 0.8431074 0.9013380 0.9882324
## Marathon 0.6764873 0.7211157 0.7126823 0.8069657 0.8777788 0.9441466
##             10000m  Marathon
## 100m     0.7147921 0.6764873
## 200m     0.7479519 0.7211157
## 400m     0.7657481 0.7126823
## 800m     0.8431074 0.8069657
## 1500m    0.9013380 0.8777788
## 5000m    0.9882324 0.9441466
## 10000m   1.0000000 0.9541630
## Marathon 0.9541630 1.0000000
```

```r
scalemen <- scale(men[,-1], T, T)
Rmen <- cor(scalemen)

loadingmen <- eigen(Rmen)$vectors
rownames(loadingmen) <- colnames(scalemen)
loadingmen
```

```
##                [,1]        [,2]         [,3]        [,4]        [,5]
## 100m     -0.3323877 -0.52939911  0.343859303  0.38074525 -0.29967117
## 200m     -0.3460511 -0.47039050 -0.003786104  0.21702322  0.54143422
## 400m     -0.3391240 -0.34532929 -0.067060507 -0.85129980 -0.13298631
## 800m     -0.3530134  0.08945523 -0.782711152  0.13427911  0.22728254
## 1500m    -0.3659849  0.15365241 -0.244270040  0.23302034 -0.65162403
## 5000m    -0.3698204  0.29475985  0.182863147 -0.05462441 -0.07181636
## 10000m   -0.3659489  0.33360619  0.243980694 -0.08706927  0.06133263
## Marathon -0.3542779  0.38656085  0.334632969  0.01812115  0.33789097
##                 [,6]       [,7]         [,8]
## 100m     -0.36203713  0.3476470 -0.065701445
## 200m      0.34859224 -0.4398969  0.060755403
## 400m      0.07708385  0.1135553 -0.003469726
## 800m     -0.34130845  0.2588830 -0.039274027
## 1500m     0.52977961 -0.1470362 -0.039745509
## 5000m    -0.35914382 -0.3283202  0.705684585
## 10000m   -0.27308617 -0.3511133 -0.697181715
## Marathon  0.37516986  0.5941571  0.069316891
```

```r
eigenmen <- eigen(Rmen)$values
eigenmen
```

```
## [1] 6.703289951 0.638410110 0.227524494 0.205849181 0.097577441 0.070687912
## [7] 0.046942050 0.009718862
```

```r
sum(eigenmen)
```

```
## [1] 8
```

Comment:

The correlation matrix before and after the standardization should be the same!!!

The eigevenctors here are called “loadings.” (it tells me the direction) It should be the same whatever ways I get upto the **sign difference.**

Eigenvalues are useful in determining proportion of variation. (it tells me the signicance of the direction)

The sum of eigenvalues are equal to the numberof columns. (it is 7 since the first column is just name of the countries)

Part b - Men

```r
loadingmen[,1:2]
```

```
##                [,1]        [,2]
## 100m     -0.3323877 -0.52939911
## 200m     -0.3460511 -0.47039050
## 400m     -0.3391240 -0.34532929
## 800m     -0.3530134  0.08945523
## 1500m    -0.3659849  0.15365241
## 5000m    -0.3698204  0.29475985
## 10000m   -0.3659489  0.33360619
## Marathon -0.3542779  0.38656085
```

```r
pcmen <- scalemen %*% loadingmen
colnames(pcmen) <- paste0("PC", 1:8)
rownames(pcmen) <- men[,1]

head(pcmen[,1:2])
```

```
##                  PC1        PC2
## Argentina  0.4163326 -0.3945394
## Australia  2.3525022  0.5502192
## Austria    0.7306318 -0.1805723
## Belgium    1.9797765 -0.3770560
## Bermuda   -1.4861338  1.6421881
## Brazil     2.2082526  0.7916572
```

```r
#Check with prcomp
#head(prcomp(men[,-1], scale = T)$x[,1:2])




eigenmen[1:2]
```

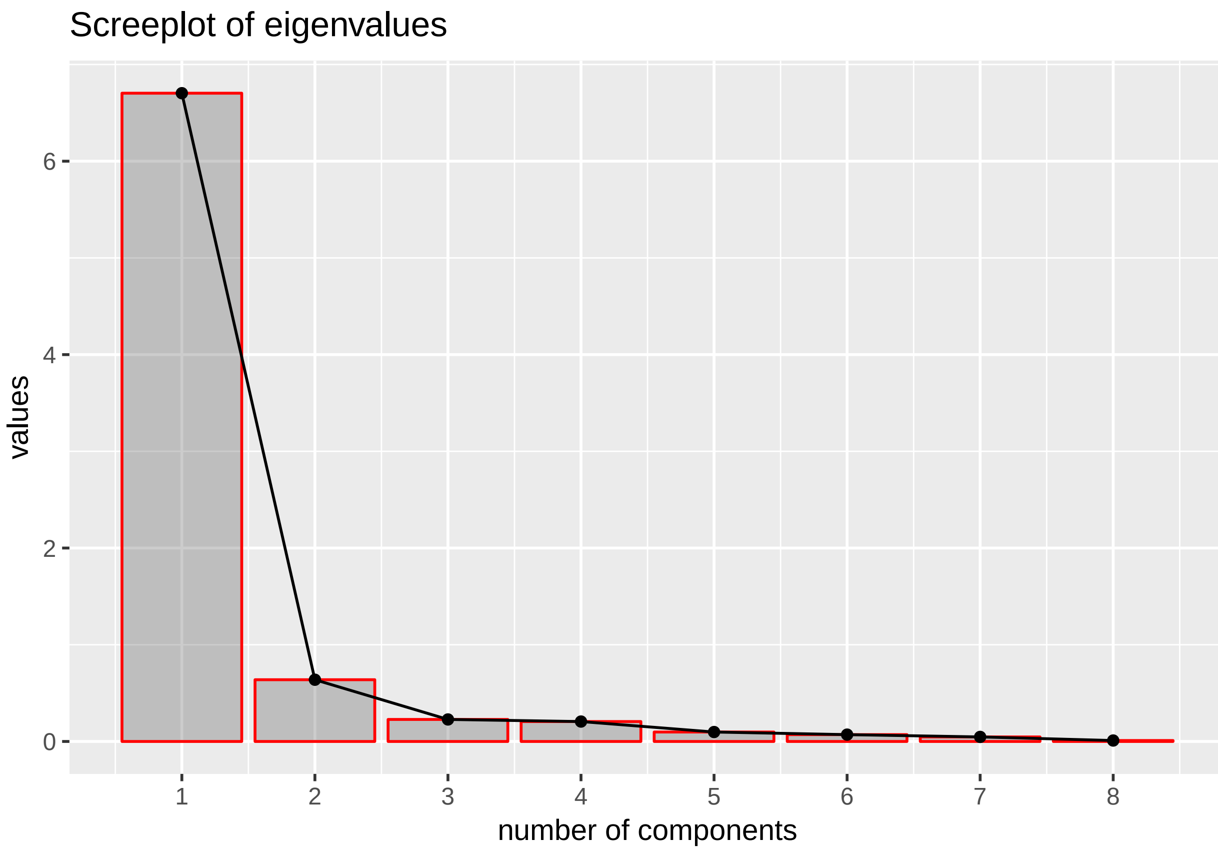
```
## [1] 6.7032900 0.6384101
```

```r
#Check with procomp
#as.vector((prcomp(men[,-1], scale = T)$sdev)^2)[1:2]
```

```r
#Table of variance explained
eigen_data <- matrix(0, nrow = round(sum(eigenmen),0), ncol = 3)
colnames(eigen_data) <- c("eigenvalue", "percentage", "cumulative.percentage")
rownames(eigen_data) <- paste0("comp", 1:sum(eigenmen))

eigen_data[,1] <- eigenmen
percentage <- apply(as.matrix(eigenmen), 2, sum(eigenmen), FUN = "/") * 100
eigen_data[,2] <- percentage

cum_fun <- function(x){ #x should be n * 1 column matrix
  for (i in 2:nrow(x)){
    x[i,] <- x[i-1,] + x[i,]
  }
  return(x)
}
cumulative <- cum_fun(percentage) #or use cumsum!!!
eigen_data[,3] <- cumulative

print(eigen_data)
```

```
##        eigenvalue percentage cumulative.percentage
## comp1 6.703289951 83.7911244              83.79112
## comp2 0.638410110  7.9801264              91.77125
## comp3 0.227524494  2.8440562              94.61531
## comp4 0.205849181  2.5731148              97.18842
## comp5 0.097577441  1.2197180              98.40814
## comp6 0.070687912  0.8835989              99.29174
## comp7 0.046942050  0.5867756              99.87851
## comp8 0.009718862  0.1214858             100.00000
```

```r
graph <- ggplot(as.data.frame(eigen_data[,1]), aes(x = 1:8, y = as.numeric(eigen_data[,1])))
graph <- graph + geom_bar(stat = "identity", alpha = 0.3, color = "red") + geom_point() +
  geom_line() +
  labs(title = "Screeplot of eigenvalues", x = "number of components", y = "values") +
  scale_x_continuous(breaks=seq(1,12,1))
graph
```



Comment:

Again Z should be the same no matter which way I used, upto the sign difference. Please check my output above for my first two PCs.

Again, we need to use the formula $\frac{\lambda_i}{\lambda_1 + \ldots + \lambda_p}$ is the proportion of variance captured by i-th principal components, when i = 1, . . . , p.

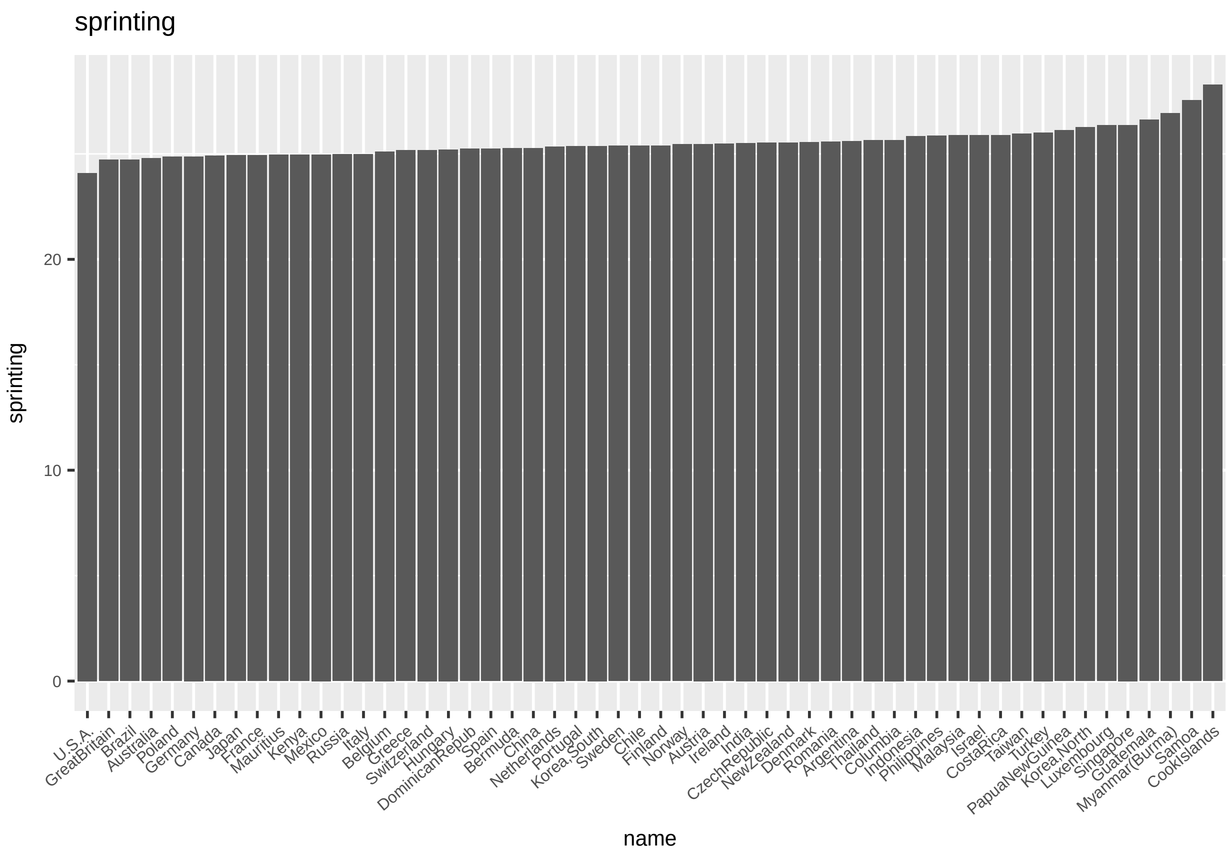
The cumulative percentage of the total sample variance explained by the two components is around 91.77 %. (similar to women's. . .)

I also made a scree-plot, which is one of the ways to choose how many PCs I should use. (Personally, I am not a fan of scree-plot, since it is too subjective. I prefer predetermined amount of variation, Kaiser's rule, or Jolife's rule.)

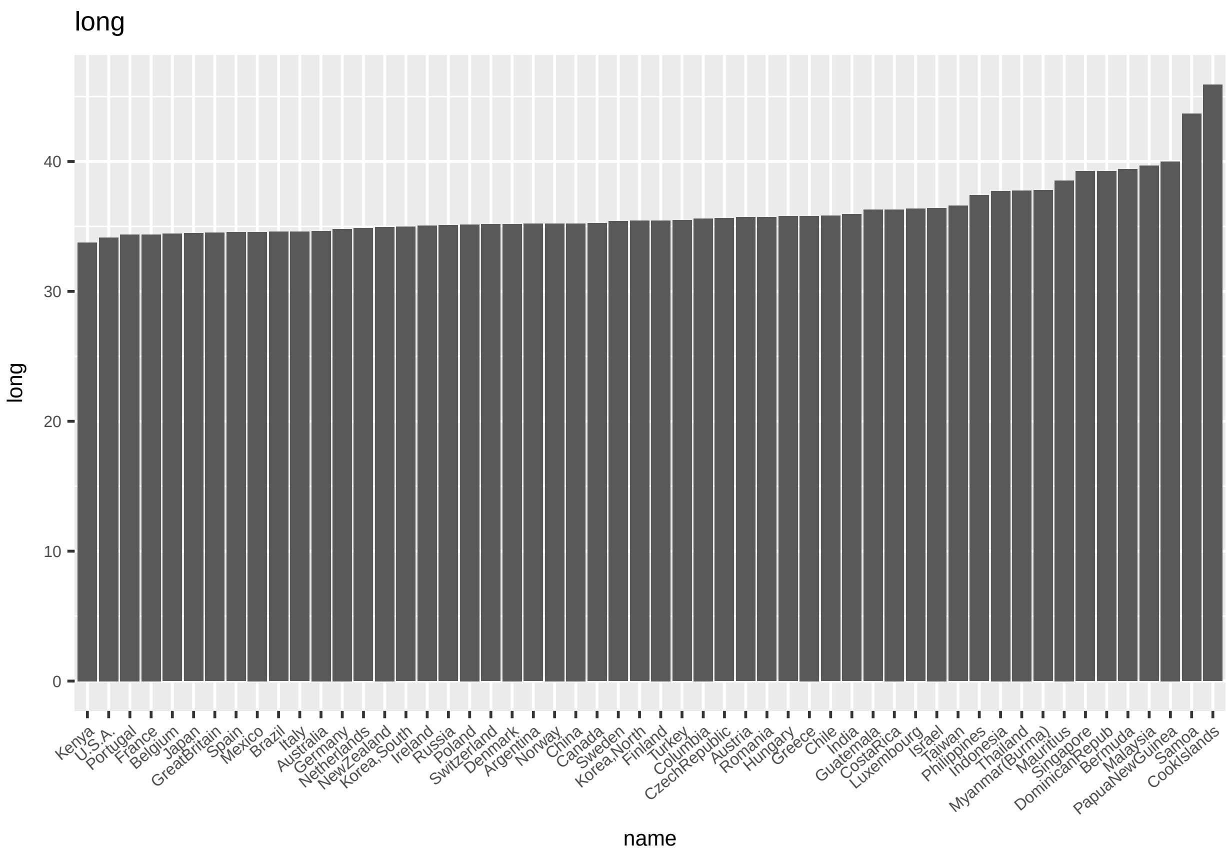
I think I will only need two dimensions of PCs for this data.

Part c - Men

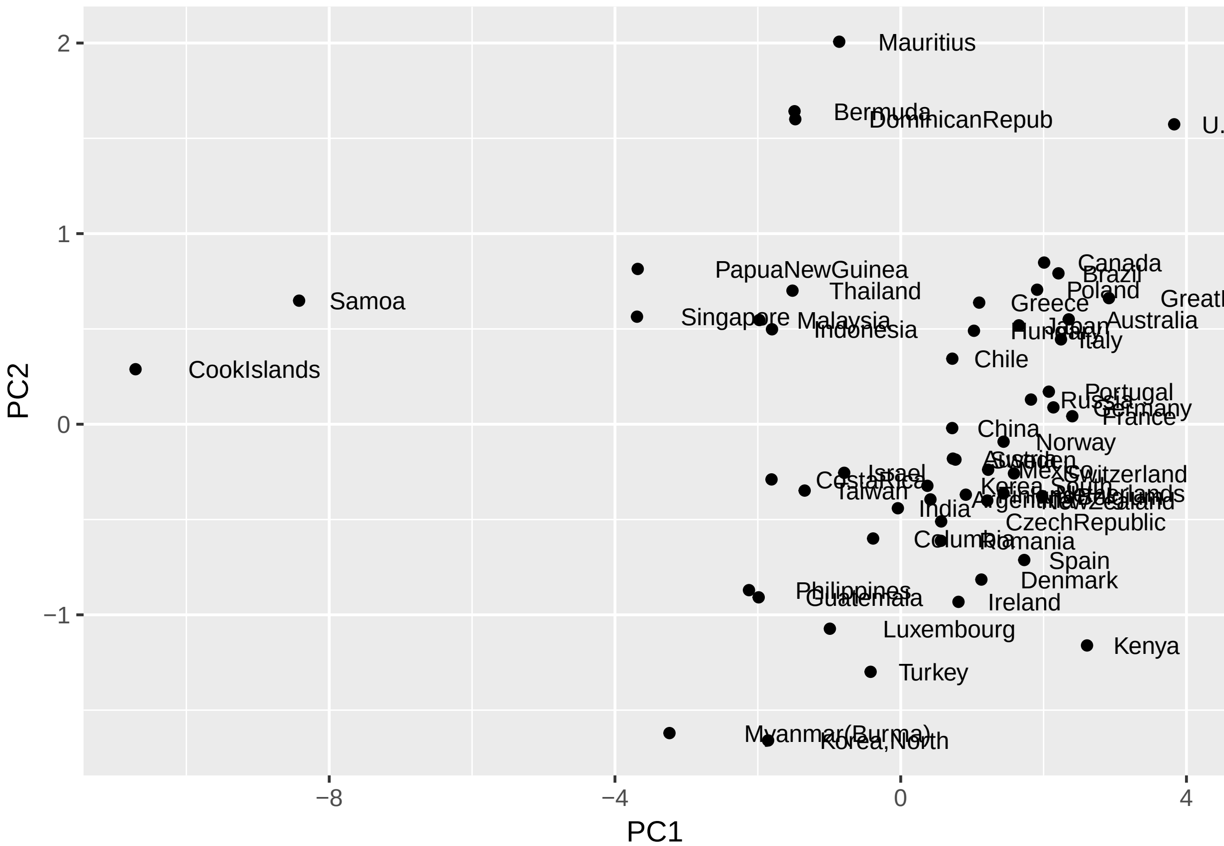
```
men$sprinting <- apply(men[,2:4], 1, mean)
men$long <- apply(men[,5:9], 1, mean)

mensprint <- data.frame(sprinting = men[,10])
menlong <- data.frame(long = men[,11])

mensprint$Rank <- rank(mensprint$sprinting)
rownames(mensprint) <- men[,1]
menlong$Rank <- rank(menlong$long)
rownames(menlong) <- men[,1]

mensprintorder <- mensprint[order(mensprint$Rank), ]
mensprintorder$name <- rownames(mensprintorder)
mensprintorder$name <- factor(mensprintorder$name,
                              levels = mensprintorder$name[order(mensprintorder$sprinting)])


menlongorder <- menlong[order(menlong$Rank), ]
menlongorder$name <- rownames(menlongorder)
menlongorder$name <- factor(menlongorder$name,
                            levels = menlongorder$name[order(menlongorder$long)])


ggplot(mensprintorder, aes(x = name, y = sprinting)) + geom_bar(stat = "identity") +
  theme(text = element_text(size=8), axis.text.x = element_text(angle = 40, hjust = 1)) +
  ggtitle("sprinting")
```



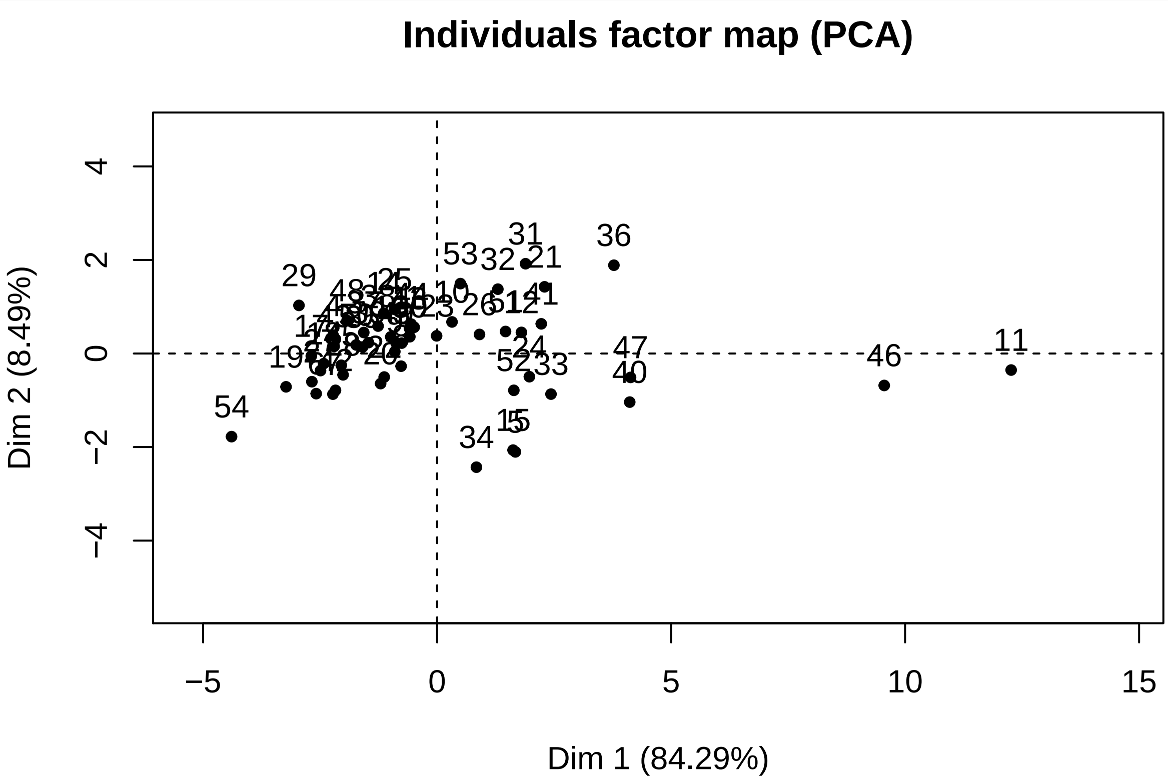

```r
ggplot(menlongorder, aes(x = name, y = long)) + geom_bar(stat = "identity") +
  theme(text = element_text(size=8), axis.text.x = element_text(angle = 40, hjust = 1)) +
  ggtitle("long")
```

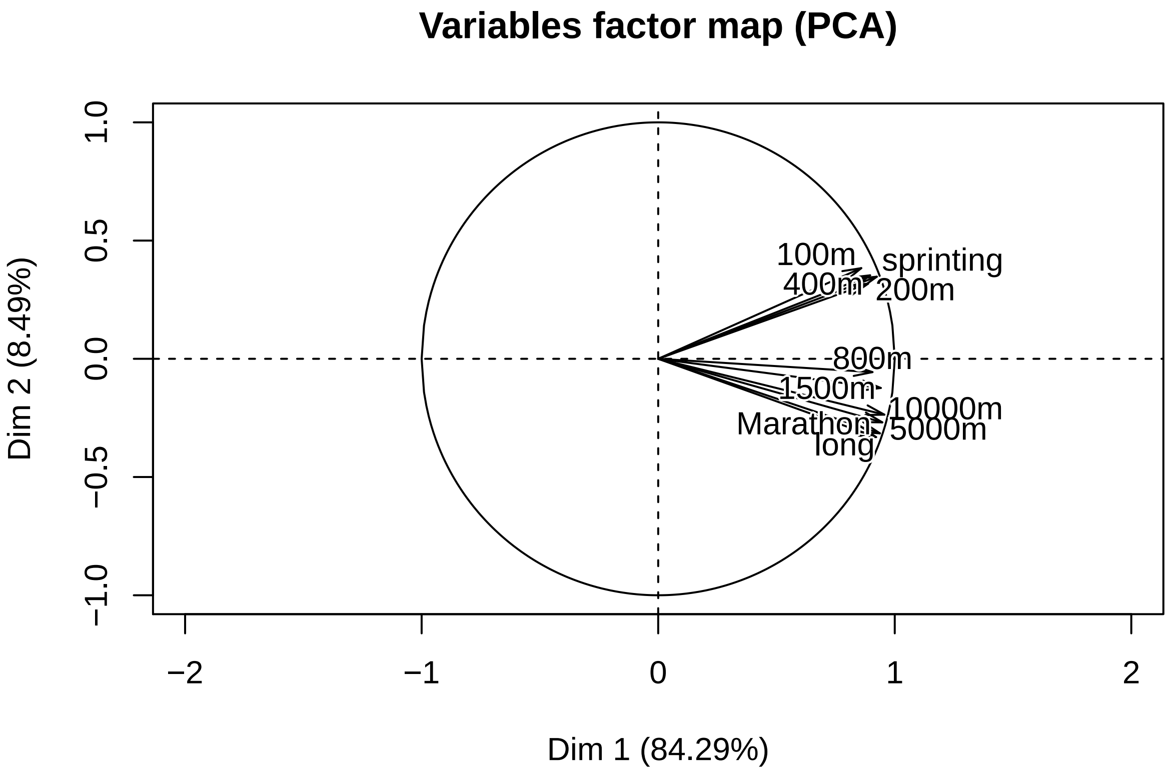


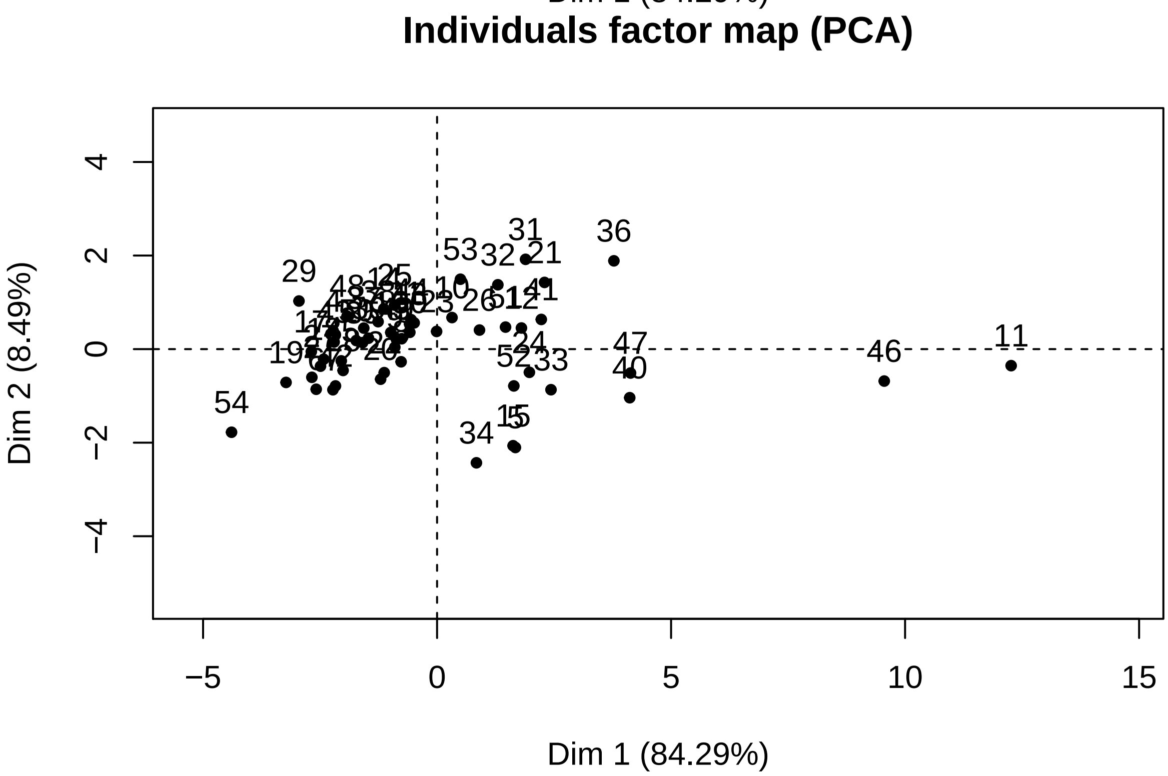

```
menpca <- prcomp(men[,-1], scale = T)
# fviz_pca_ind(menpca,
#              col.ind = "cos2", # Color by the quality of representation
#              gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
#              repel = TRUE     # Avoid text overlapping
#              )
ggplot(as.data.frame(pcmen[,1:2]), aes(x = PC1, y = PC2)) + geom_point() +
  geom_text(aes(label = rownames(pcmen), hjust = -0.4), size = 3)
```




```r
plot(PCA(men[,-1]))
```







Comment:

Please also refer to the graph I made for part d).

As you can see from my plots, the interpretation of men's and women's are really similar, and this makes sense when you refer to the extra bar plots I made. They show the similar

countries' rankings as women's.

First of all, I think by looking at the scree-plot, I feel like I only need two PCs to explain the data well enough (looking for elbow).

Second, when I see the graphs, most of the countries are left-skwed... And, as USA, UK, Canada, and Austrailia (the countries who have good amounts of athletes) are placed in the right-side, I can tell that the countries who have done pretty well in runnings are placed in the right-side of PC1. Only three countries: SAM, COK, Singapore are on the left-hand side.

I think the first component indicates how good each coungry is in the short distance (sprinting) like for example in 100m, 200m, and 400m. (I ranked the long distances and short distances.) It is quite clear as USA, UK, Canada, and Austrailia all placed in the top tiers; however, for example, KEN is not really having big number in PC1 as they are doing well in the long distances but not that well in the short distances. However, KEN does pretty well here in short distance for men, so they actually place on the right hand side in PC1 (However, in overall, PC1 shows atheletic excellence.)

The second component is not clear to be interpreted, and this makes sense as this component does not take small variance, as we saw in the previous question (eigenvalue). However, one thing I found that might be possible interpretation for PC2 shows how big the difference is betwene short and long distances running... For example, KEN and KOR.N show the good amounts of gaps between short and long runnings, but USA and UK did not... However, this is not a perfect interpretation...

PC1: **shows atheletic excellence.**

PC2: **difference is betwene short and long distances running (so how countries are good at long run compared to short run)**

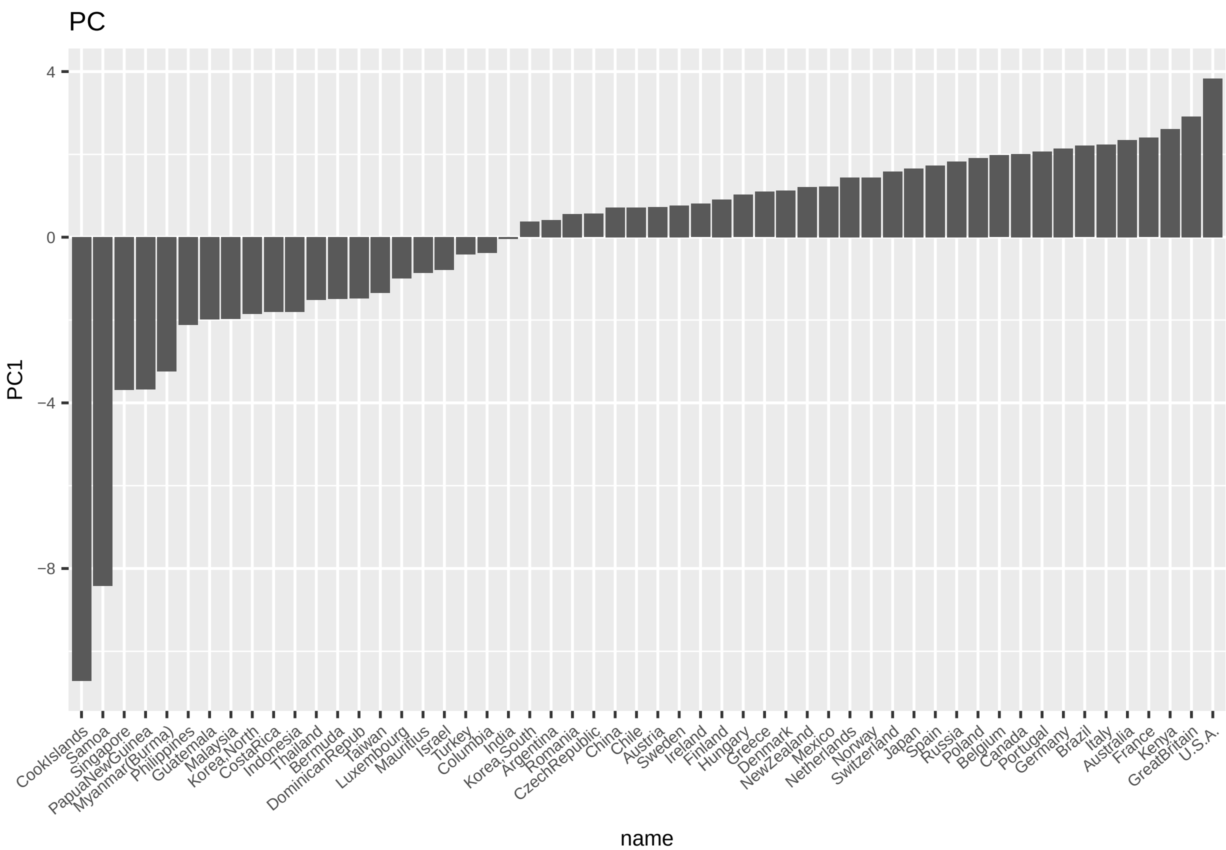
Part d - Men

```
pcmenrank <- data.frame(PC1 = pcmen[,1])
pcmenrank$Rank <- rank(pcmenrank$PC1)
order <- pcmenrank[order(pcmenrank$Rank), ]
order$name <- rownames(order)
order$name <- factor(order$name, levels = order$name[order(order$PC1)])
order
```

```
##                         PC1 Rank           name
## CookIslands    -10.71119651    1    CookIslands
## Samoa           -8.42153735    2          Samoa
## Singapore       -3.69149229    3      Singapore
## PapuaNewGuinea  -3.68134494    4 PapuaNewGuinea
## Myanmar(Burma)  -3.23684547    5 Myanmar(Burma)
## Philippines     -2.12382696    6    Philippines
## Guatemala       -1.98836313    7      Guatemala
## Malaysia        -1.97947150    8       Malaysia
```

```
## Korea,North     -1.85611284    9    Korea,North
## CostaRica       -1.80827499   10      CostaRica
## Indonesia       -1.80145079   11      Indonesia
## Thailand        -1.51481290   12       Thailand
## Bermuda         -1.48613375   13        Bermuda
## DominicanRepub  -1.47568252   14 DominicanRepub
## Taiwan          -1.34431318   15         Taiwan
## Luxembourg      -0.99164938   16     Luxembourg
## Mauritius       -0.86045396   17      Mauritius
## Israel          -0.79065356   18         Israel
## Turkey          -0.42141313   19         Turkey
## Columbia        -0.38603599   20       Columbia
## India           -0.03970824   21          India
## Korea,South      0.37364381   22    Korea,South
## Argentina        0.41633265   23      Argentina
## Romania          0.56233650   24        Romania
## CzechRepublic    0.56671042   25  CzechRepublic
## China            0.72128317   26          China
## Chile            0.72256355   27          Chile
## Austria          0.73063182   28        Austria
## Sweden           0.76690338   29         Sweden
## Ireland          0.80892204   30        Ireland
## Finland          0.91160091   31        Finland
## Hungary          1.02549631   32        Hungary
## Greece           1.09870962   33         Greece
## Denmark          1.12932147   34        Denmark
## NewZealand       1.21196841   35     NewZealand
## Mexico           1.22404745   36         Mexico
## Netherlands      1.43961748   37    Netherlands
## Norway           1.43996309   38         Norway
## Switzerland      1.58538872   39    Switzerland
## Japan            1.65463811   40          Japan
## Spain            1.72965172   41          Spain
## Russia           1.82365926   42         Russia
## Poland           1.90966469   43         Poland
## Belgium          1.97977654   44        Belgium
## Canada           2.00766173   45         Canada
## Portugal         2.07397725   46       Portugal
## Germany          2.13786396   47        Germany
## Brazil           2.20825258   48         Brazil
## Italy            2.24390003   49          Italy
## Australia        2.35250215   50      Australia
## France           2.40202950   51         France
## Kenya            2.60834729   52          Kenya
## GreatBritain     2.91498277   53   GreatBritain
## U.S.A.           3.82842499   54         U.S.A.
```

```r
ggplot(order, aes(x = name, y = PC1)) + geom_bar(stat = "identity") +
  theme(text = element_text(size=8), axis.text.x = element_text(angle = 40, hjust = 1)) +
  ggtitle("PC")
```



Comment:

As I commented in the previous question, I think this ranking shows the notion of athletic excellence in the **overall sprinting and long running. (little bit different with Women's) - but still I think PC1 takes short distance with more weights** Most of the countires who have high PC here have good records in short distance runnings. USA, UK, France, Austrailia, and Kenya are all doing well on the short distance (But most of them are also doing pretty well in long distance as well).

Are the results consistent with those obtained from the women's data?

Comment:

The results are mostly consistent with the ones obtained from the women's data!!! And, this makes sense actually. Atheletic performance usually are not that different based on geneder.

For women's data, USA, Germany, Russia, China, and France are on the top five; however, for men's data, USA, UK, Kenya, France, and Austrailia are on the top five.

Part e - Men

```
men1 <- 100 / men[,2]
men2 <- 200 / men[,3]
men3 <- 400 / men[,4]
men4 <- 800 / (men[,5] * 60)
men5 <- 1500 / (men[,6] * 60)
men6 <- 5000 / (men[,7] * 60)
men7 <- 10000 / (men[,8] * 60)
men8 <- 42195 / (men[,9] * 60)
mene <- data.frame(`100m` = men1, `200m` = men2, `400m` = men3, `800m` = men4,
                   `1500m` = men5, `5000m` = men6, `10000m` = men7, marathon = men8)


#A
cov(mene)
```

```
##               X100m      X200m      X400m      X800m     X1500m     X5000m
## X100m    0.04349790 0.04827718 0.04346323 0.03149513 0.04250343 0.04692523
## X200m    0.04827718 0.06484523 0.05586780 0.04323338 0.05352645 0.05877310
## X400m    0.04346323 0.05586780 0.06882169 0.04282214 0.05372066 0.06176643
## X800m    0.03149513 0.04323338 0.04282214 0.04688400 0.05230584 0.05715598
## X1500m   0.04250343 0.05352645 0.05372066 0.05230584 0.07291400 0.07663884
## X5000m   0.04692523 0.05877310 0.06176643 0.05715598 0.07663884 0.09593980
## X10000m  0.04483253 0.05725123 0.05993536 0.05539454 0.07457187 0.09373567
## marathon 0.04312562 0.05629446 0.05673423 0.05419108 0.07365179 0.09058189
##             X10000m   marathon
## X100m    0.04483253 0.04312562
## X200m    0.05725123 0.05629446
## X400m    0.05993536 0.05673423
## X800m    0.05539454 0.05419108
## X1500m   0.07457187 0.07365179
## X5000m   0.09373567 0.09058189
## X10000m  0.09428944 0.09099518
## marathon 0.09099518 0.09792763
```

```
covmen <- cov(scale(mene, T, F)) #should be the same!!!

loadingmene <- eigen(covmen)$vectors
rownames(loadingmene) <- colnames(mene)
-1 * loadingmene # I multipled by -1 just to make the first column to be positive...
```

```
##                [,1]        [,2]        [,3]        [,4]        [,5]
```

```
## X100m    0.2439701  0.43237108 -0.1727480 -0.44964279  0.39041927
## X200m    0.3113827  0.52345617 -0.2352392 -0.31840509 -0.34122401
## X400m    0.3168151  0.46905827  0.6843621  0.41957963 -0.04645807
## X800m    0.2775048  0.03280175 -0.4363747  0.54293696 -0.33164606
## X1500m   0.3642621 -0.06284374 -0.4386419  0.31687475  0.30302349
## X5000m   0.4276861 -0.26134677  0.1112433 -0.01627787  0.37446287
## X10000m  0.4209180 -0.30988613  0.1869193 -0.09987801  0.21458833
## marathon 0.4163706 -0.38688033  0.1277716 -0.33906010 -0.58386443
##                [,6]        [,7]         [,8]
## X100m    -0.1192313  0.58379960 -0.118524362
## X200m     0.2468224 -0.53450922  0.096228396
## X400m    -0.1773787  0.03913497 -0.007800701
## X800m     0.3682149  0.43152617 -0.070182724
## X1500m   -0.6080110 -0.32723811 -0.044294597
## X5000m    0.3335143 -0.00576378  0.696171699
## X10000m   0.3517319 -0.18028526 -0.692544534
## marathon -0.3913984  0.21473415  0.073963688
```

```r
eigenmene <- eigen(covmen)$values
eigenmene
```

```
## [1] 0.494049954 0.046223803 0.013912284 0.013320803 0.007522548 0.005749212
## [7] 0.003220375 0.001120710
```

```r
#B
-1 * loadingmene[,1:2]
```

```
##               [,1]        [,2]
## X100m    0.2439701  0.43237108
## X200m    0.3113827  0.52345617
## X400m    0.3168151  0.46905827
## X800m    0.2775048  0.03280175
## X1500m   0.3642621 -0.06284374
## X5000m   0.4276861 -0.26134677
## X10000m  0.4209180 -0.30988613
## marathon 0.4163706 -0.38688033
```

```r
pcmene <- scale(mene, T, F) %*% loadingmene
colnames(pcmene) <- paste0("PC", 1:8)
rownames(pcmene) <- men[,1]

head(pcmene[,1:2])
```

```
##                  PC1        PC2
## Argentina -0.1478610  0.1356036
## Australia -0.6499347 -0.1298034
## Austria   -0.2090035  0.0624355
## Belgium   -0.6115287  0.1120454
## Bermuda    0.5667930 -0.5089854
## Brazil    -0.5459834 -0.2474470
```

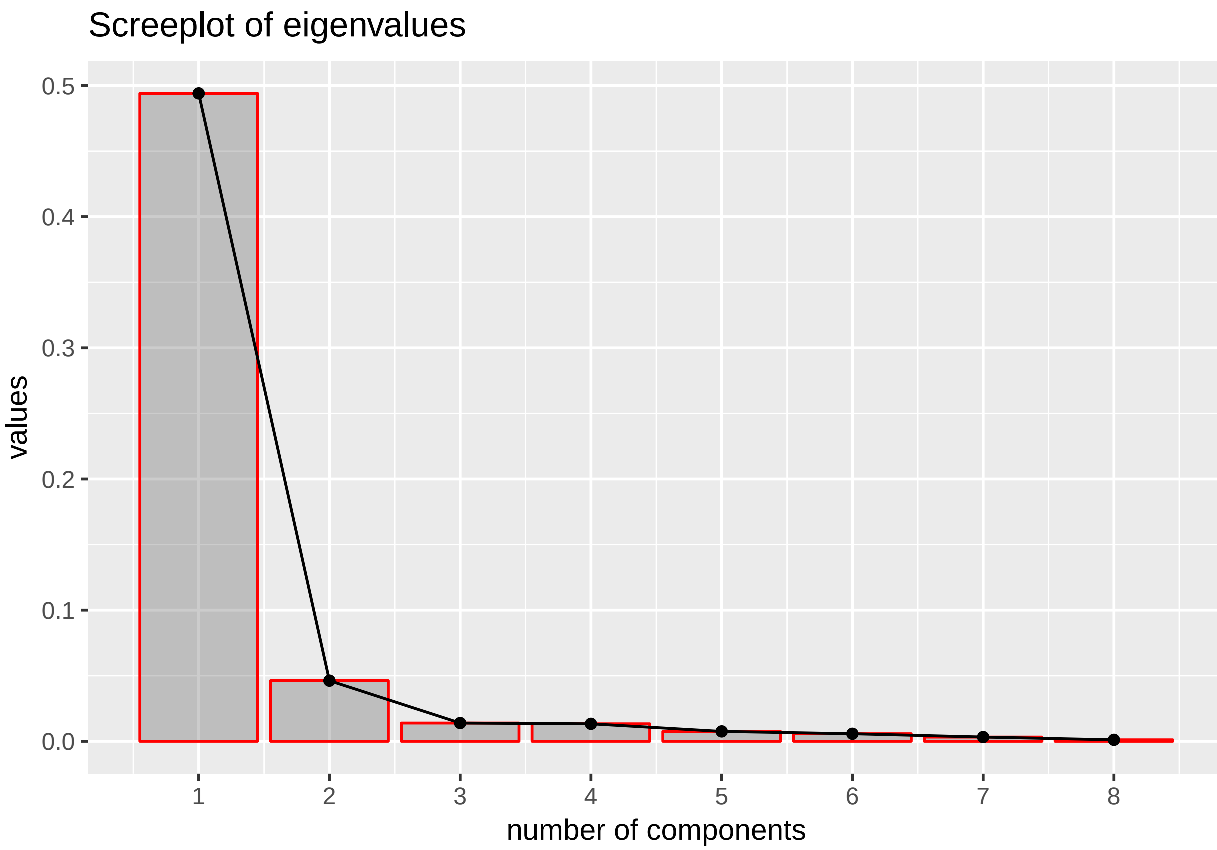
```r
#Check with prcomp
#head(prcomp(mene, center = T)$x[,1:2])

eigenmene[1:2]
```

```
## [1] 0.4940500 0.0462238
```

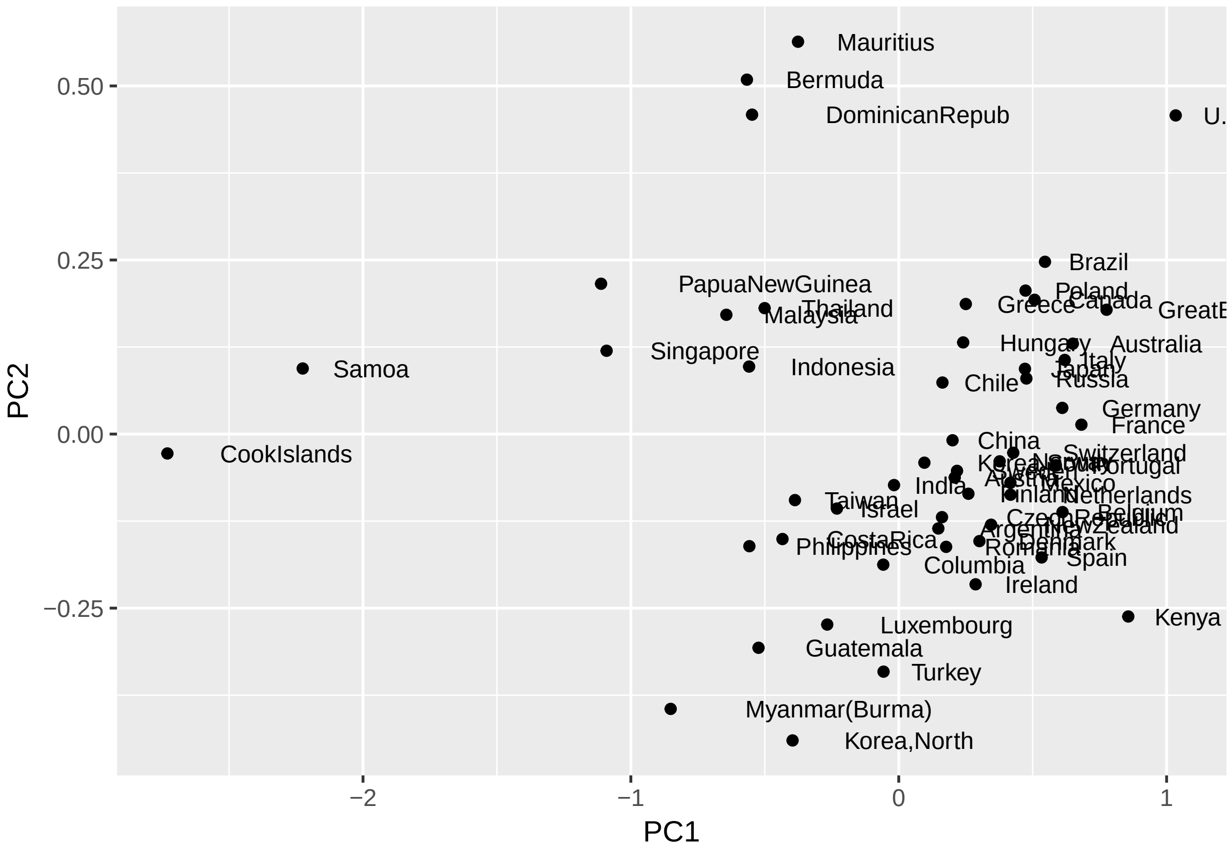
```r
#Check with procomp
#as.vector((prcomp(mene, center = T)$sdev)^2)[1:2]

#Table of variance explained
eigen_data <- matrix(0, nrow = 8, ncol = 3)
colnames(eigen_data) <- c("eigenvalue", "percentage", "cumulative.percentage")
rownames(eigen_data) <- paste0("comp", 1:8)

eigen_data[,1] <- eigenmene
percentage <- apply(as.matrix(eigenmene), 2, sum(eigenmene), FUN = "/") * 100
eigen_data[,2] <- percentage

cum_fun <- function(x){ #x should be n * 1 column matrix
  for (i in 2:nrow(x)){
    x[i,] <- x[i-1,] + x[i,]
  }
  return(x)
}
cumulative <- cum_fun(percentage) #or use cumsum!!!
eigen_data[,3] <- cumulative

print(eigen_data)
```

```
##        eigenvalue percentage cumulative.percentage
## comp1 0.494049954 84.4357083              84.43571
## comp2 0.046223803  7.8998884              92.33560
## comp3 0.013912284  2.3776818              94.71328
## comp4 0.013320803  2.2765945              96.98987
## comp5 0.007522548  1.2856426              98.27552
## comp6 0.005749212  0.9825703              99.25809
## comp7 0.003220375  0.5503789              99.80846
## comp8 0.001120710  0.1915352             100.00000
```

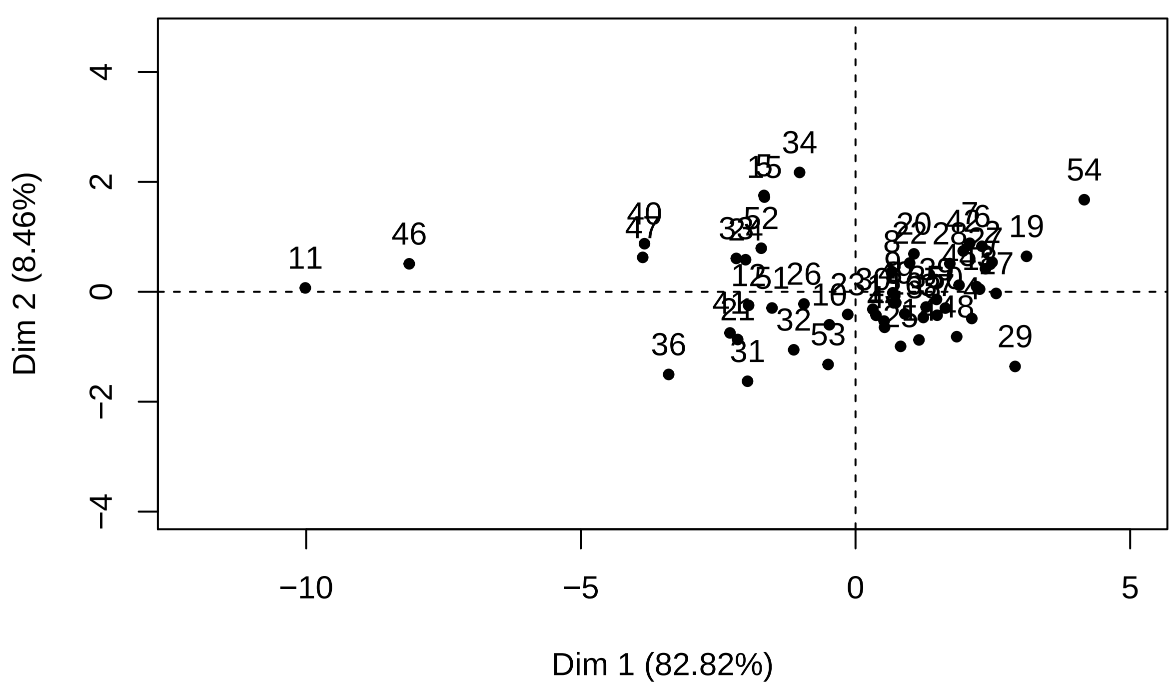
```r
graph <- ggplot(as.data.frame(eigen_data[,1]), aes(x = 1:8, y = as.numeric(eigen_data[,1])))
graph <- graph + geom_bar(stat = "identity", alpha = 0.3, color = "red") + geom_point() +
  geom_line() +
  labs(title = "Screeplot of eigenvalues", x = "number of components", y = "values") +
  scale_x_continuous(breaks=seq(1,12,1))
graph
```

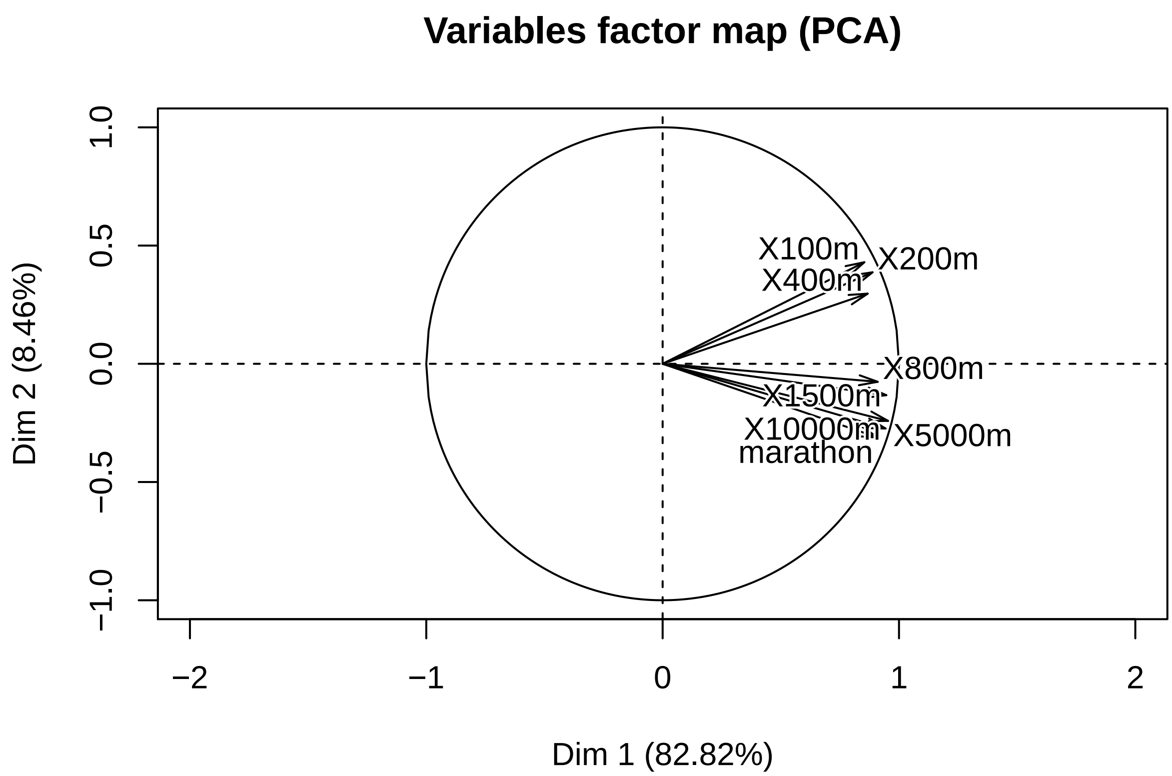


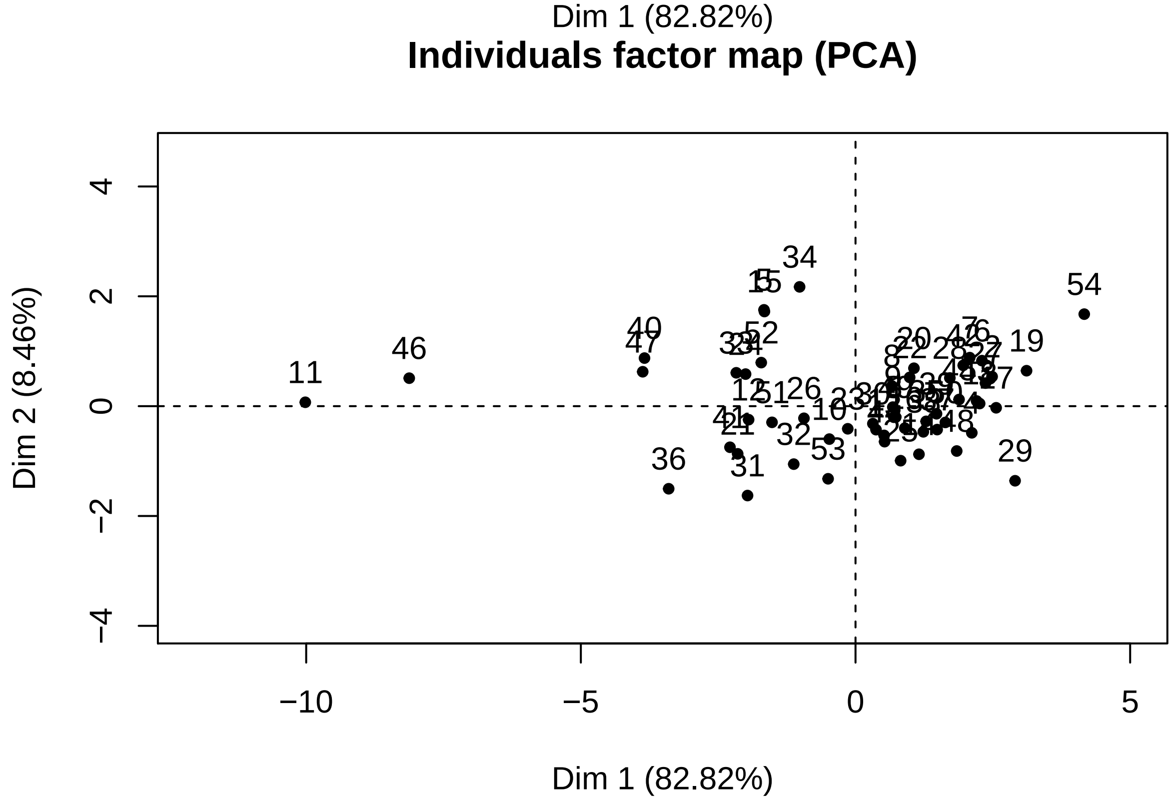

```
#C
menpcae <- prcomp(mene, center = T)
# fviz_pca_ind(menpcae,
#              col.ind = "cos2", # Color by the quality of representation
#              gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
#              repel = TRUE     # Avoid text overlapping
#              )
ggplot(as.data.frame(-1 *pcmene[,1:2]), aes(x = PC1, y = PC2)) + geom_point() +
  geom_text(aes(label = rownames(pcmene), hjust = -0.4), size = 3)
```




```
plot(PCA(mene))
```







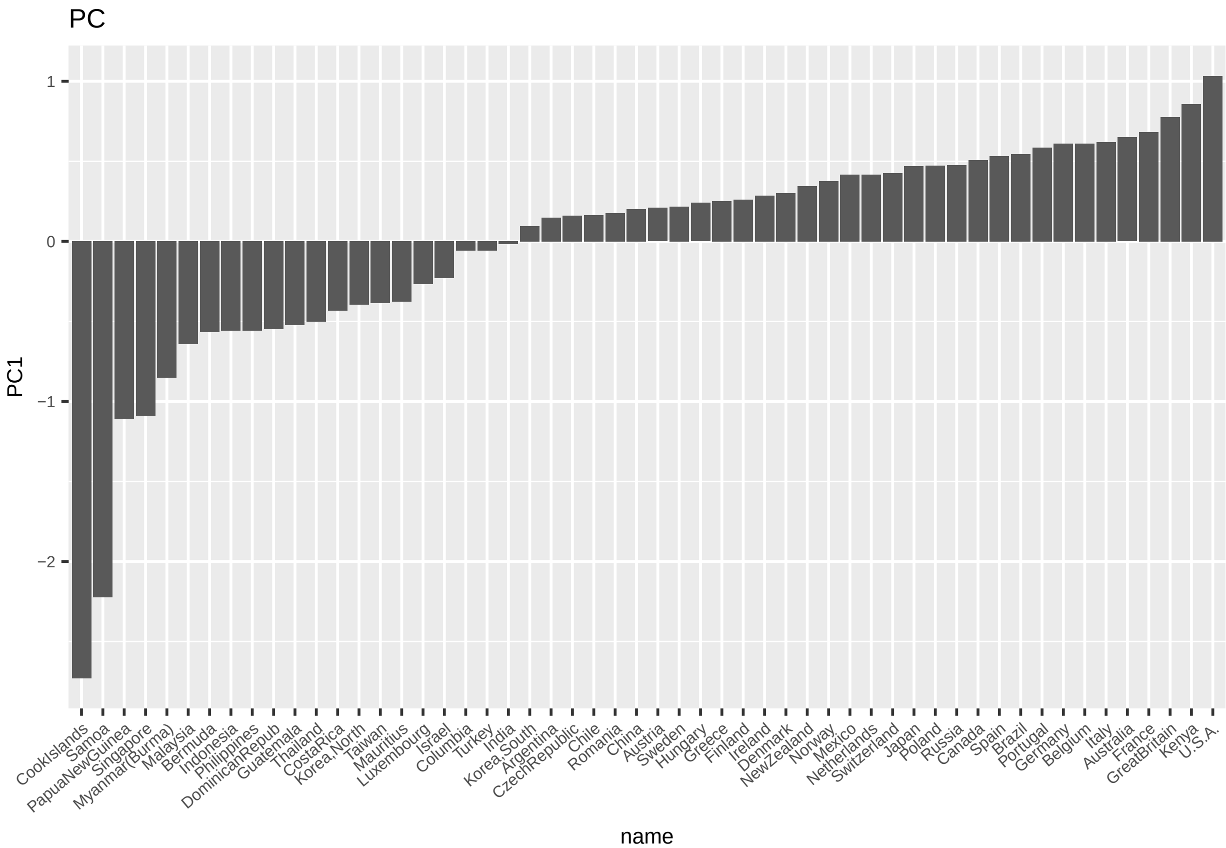

```
#D
pcmenranke <- data.frame(PC1 = -1 * pcmene[,1])
pcmenranke$Rank <- rank(pcmenranke$PC1)
order <- pcmenranke[order(pcmenranke$Rank), ]
order$name <- rownames(order)
```

```r
order$name <- factor(order$name, levels = order$name[order(order$PC1)])
order
```

```
##                       PC1 Rank           name
## CookIslands    -2.73021554    1    CookIslands
## Samoa          -2.22501621    2          Samoa
## PapuaNewGuinea -1.11125588    3 PapuaNewGuinea
## Singapore      -1.09066120    4      Singapore
## Myanmar(Burma) -0.85135709    5 Myanmar(Burma)
## Malaysia       -0.64349447    6       Malaysia
## Bermuda        -0.56679298    7        Bermuda
## Indonesia      -0.55854132    8      Indonesia
## Philippines    -0.55759756    9    Philippines
## DominicanRepub -0.54750955   10 DominicanRepub
## Guatemala      -0.52340102   11      Guatemala
## Thailand       -0.50060854   12       Thailand
## CostaRica      -0.43398784   13      CostaRica
## Korea,North    -0.39630596   14    Korea,North
## Taiwan         -0.38742383   15         Taiwan
## Mauritius      -0.37600568   16      Mauritius
## Luxembourg     -0.26690921   17     Luxembourg
## Israel         -0.23084081   18         Israel
## Columbia       -0.05773185   19       Columbia
## Turkey         -0.05669944   20         Turkey
## India          -0.01754668   21          India
## Korea,South     0.09577359   22    Korea,South
## Argentina       0.14786099   23      Argentina
## CzechRepublic   0.16155296   24  CzechRepublic
## Chile           0.16317724   25          Chile
## Romania         0.17692856   26        Romania
## China           0.20088565   27          China
## Austria         0.20900348   28        Austria
## Sweden          0.21759478   29         Sweden
## Hungary         0.24056272   30        Hungary
## Greece          0.25043281   31         Greece
## Finland         0.26009625   32        Finland
## Ireland         0.28699863   33        Ireland
## Denmark         0.30126384   34        Denmark
## NewZealand      0.34487921   35     NewZealand
## Norway          0.37677990   36         Norway
## Mexico          0.41688080   37         Mexico
## Netherlands     0.41732532   38    Netherlands
## Switzerland     0.42755291   39    Switzerland
## Japan           0.47118701   40          Japan
## Poland          0.47329629   41         Poland
## Russia          0.47652587   42         Russia
## Canada          0.50710457   43         Canada
## Spain           0.53349369   44          Spain
## Brazil          0.54598342   45         Brazil
## Portugal        0.58646683   46       Portugal
## Germany         0.61055320   47        Germany
## Belgium         0.61152873   48        Belgium
## Italy           0.61969309   49          Italy
## Australia       0.64993465   50      Australia
```

```
## France           0.68186975   51         France
## GreatBritain     0.77582606   52   GreatBritain
## Kenya            0.85692345   53          Kenya
## U.S.A.           1.03396643   54         U.S.A.
```

```r
ggplot(order, aes(x = name, y = PC1)) + geom_bar(stat = "identity") +
  theme(text = element_text(size=8), axis.text.x = element_text(angle = 40, hjust = 1)) +
  ggtitle("PC")
```



Comment:

This is not required problem. I just did it for fun and learing.

As professor recomended, I will use mean-centered (not standardized) for covariance matrix, and standardized matrix for correlation matrix.

First of all, I want to mention I adjusted the sign, for interpretation and visual purpose. In PCA, interpretation and visual are both really important, so I was extra careful about them. . .

Definitely, as I used the covaraince matrix, the eigenvalues are different;however, the cumulative percentages are almost the same/similar. And, the two principal components are different for sure.

As you can easily see from my bar plots, the nations' ranks based on the scores on the first principal component, the bar plot looks almost the same! Thus, the rankings are not significantly different.

Furthermore, the interpretation of the components are also the same. It can be easily found on the PC1 v.s. PC2 plots.

Problem 2

Good ref: https://web.stanford.edu/class/psych253/tutorials/FactorAnalysis.html

Data import

```
pollution <- read.delim("Data-HW4-pollution.dat", header = F, sep = "", na.strings = "")

colnames(pollution) <- c("Wind", "SolarRadiation", "CO", "NO", "NO2", "O3", "HC")
dim(pollution)
```

```
## [1] 42  7
```

Part a

Using all 7 air-pollution variables to generate the sample covariance matrix.

```
cov(pollution)
```

```
##                      Wind SolarRadiation         CO         NO        NO2
## Wind            2.5000000     -2.7804878 -0.3780488 -0.4634146 -0.5853659
## SolarRadiation -2.7804878    300.5156794  3.9094077 -1.3867596  6.7630662
## CO             -0.3780488      3.9094077  1.5220674  0.6736353  2.3147503
## NO             -0.4634146     -1.3867596  0.6736353  1.1823461  1.0882695
## NO2            -0.5853659      6.7630662  2.3147503  1.0882695 11.3635308
## O3             -2.2317073     30.7909408  2.8217189 -0.8106852  3.1265970
## HC              0.1707317      0.6236934  0.1416957  0.1765389  1.0441347
##                        O3        HC
## Wind           -2.2317073 0.1707317
## SolarRadiation 30.7909408 0.6236934
## CO              2.8217189 0.1416957
```

```
## NO              -0.8106852 0.1765389
## NO2              3.1265970 1.0441347
## O3              30.9785134 0.5946574
## HC               0.5946574 0.4785134
```

```
cov <- cov(scale(pollution, T, F)) #Should be the same!!!
#cov
```

Part b

Obtain the principal component solution to a factor model with m = 1 and m = 2. Find the corresponding commonalities.

```
options(scipen=999) #eliminate the scientific notation

fit <- eigen(cov)
v <- fit$vectors
rownames(v) <- colnames(pollution)

L1 <- v[,1] * sqrt(fit$values[1]) #eigen value is already ordered automatically in R...
L2 <- v[,2] * sqrt(fit$values[2])

#One factor model
L1
```

```
##          Wind SolarRadiation             CO             NO            NO2
##    0.17511443   -17.32436626    -0.24528879     0.08215953    -0.42309094
##            O3             HC
##   -1.96110754    -0.04083029
```

```
L1^2
```

```
##          Wind SolarRadiation             CO             NO            NO2
##   0.030665064   300.133666359    0.060166589    0.006750188    0.179005941
##            O3             HC
##   3.845942794    0.001667112
```

```
#diag(L1 %*% t(L1)) -> same answer as above....


#Two factor model
cbind(L1 = -1 * L1, L2)
```

```
##                          L1          L2
## Wind            -0.17511443  0.40532535
## SolarRadiation 17.32436626  0.61765845
```

```
## CO                0.24528879 -0.52945432
## NO               -0.08215953  0.07021387
## NO2               0.42309094 -0.79965586
## O3                1.96110754 -5.17586403
## HC                0.04083029 -0.12666596
```

```r
L1^2 + L2^2 #diagonal entries of LL^T
```

```
##          Wind SolarRadiation             CO             NO            NO2
##    0.19495370   300.51516832     0.34048847     0.01168017     0.81845543
##            O3             HC
##   30.63551120     0.01771138
```

```r
#diag(cbind(L1, L2) %*% t(cbind(L1, L2))) #LL^T -> same answer as above....


#cov - cbind(L1, L2) %*% t(cbind(L1, L2)) #psi...
```

Comment:

Communality for m = 1 and m = 2 (the proportion of variance of variables that is contributed by m common factors) were printed above. The higher commonality is, the better the variable is explained by the factors. I can see that solar radiation is explained well by one factor model, and the wind, O3, CO, and NO2 are explained well by the second factor.

Remember that the method I am using is PC.

Part c

Find the proportion of variation accounted for by the one-factor model, and the two-factor model, respectively.

In the lecture note, it says that proportion of total variation due to the i-th factor is $\frac{l_{1i}^2 + ... + l_{pi}^2}{\sigma_1 + ... + \sigma_p}$, where σ_j is the diagonal element from covariance matrix.

If I used the correlation matrix sum of all the simgas are just length of L.

```r
options(scipen=999)

sum(L1^2) / sum(diag(cov)) #one factor model
```

```
## [1] 0.872948
```

```r
sum(L1^2 + L2^2) / sum(diag(cov)) #two factor model
```

```
## [1] 0.9540751
```

Comment:

The variance indicates the variability in the data explained by each factor.

Definitely as I used the model with more factors, the more proportion of variation will be accounted for. Around 0.873 is accounted by the one factor model, and around 0.954 by the two factor model.

Part d

Perform a varimax rotation of the m = 2 solution, and interpret the factors after the rotation. Find the proportion of variation accounted for by the two-factor model after the rotation.

```r
options(scipen=999)

varimax(cbind(L1, L2), normalize = F)
```

```
## $loadings
##
## Loadings:
##                L1      L2
## Wind             0.157   0.413
## SolarRadiation -17.335  -0.133
## CO              -0.222  -0.540
## NO
## NO2             -0.388  -0.817
## O3              -1.735  -5.256
## HC                      -0.128
##
##                     L1     L2
## SS loadings    303.740 28.794
## Proportion Var  43.391  4.113
## Cumulative Var  43.391 47.505
##
## $rotmat
##             [,1]       [,2]
## [1,]  0.99906181 0.04330701
## [2,] -0.04330701 0.99906181
```

Comment:

Factor rotation simplifies the loading structure and makes the factors be better distinguishable, which eventually helps us to interpret. When I use the varimax() function in R, they automatically get rid of the elements from the variable where the factor barely has influence on. The factor 1 has the most influence on solar radiation, so the factor 1 describes solar radiation-related issue/pollution. The factor 2 has the significant influence on O3 (and NO2 slightly). So the factor 2 describes ozone-related issue/pollution.

Proportion of variations accounted for by the two-factor model are around 43.391 and 4.113 as you can see from the output above. **(47.505% for m = 2)**

We have used covariance for the factor analysis, but I personally preferred to do it with correlation matrix as the loading ranges from -1 to 1 here. . .