Homework 1 (Coding Part) for Stat 154

Kevin Luo

Sept 5, 2019

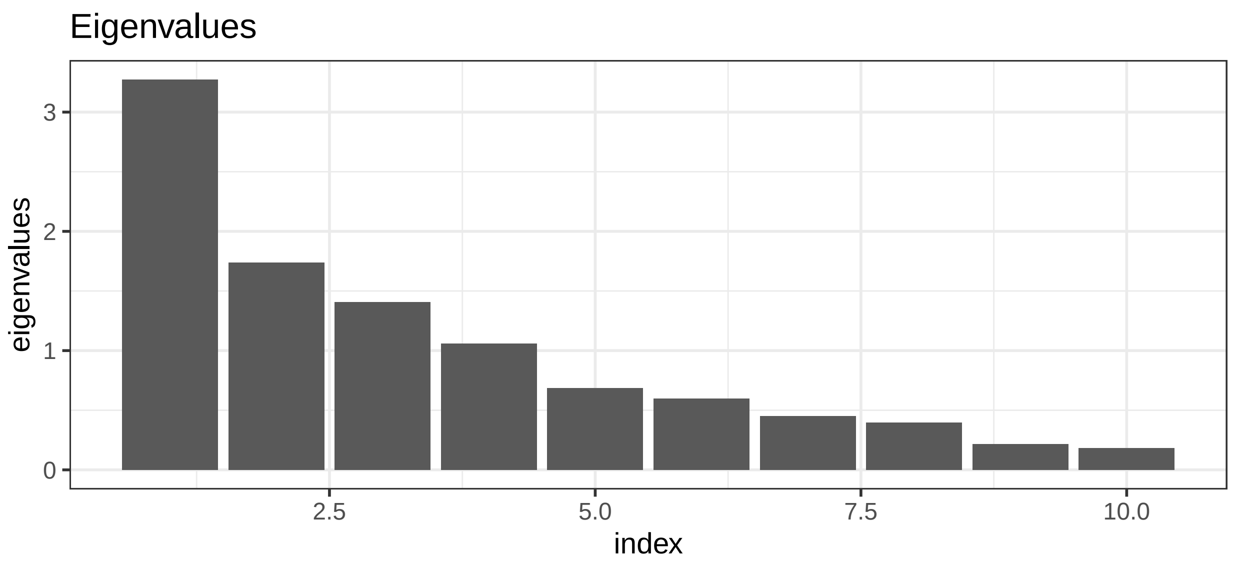
Problem 5

5.1 Should we run PCA on transformed data?
Yes! Note that here we have different measures of different variables. (Some are measured in time and others in height, etc.) It is probably wise to scale the data first. (i.e. PCA on the basis of correlation matrix)
Rmk: If the variables are measured on the same scale, it's fair not to scale the data as it's plausible to keep those with more variance que sera sera.

```r
# The code shall be:
pca <- PCA(X = dat, scale.unit = TRUE, ncp = 10, graph = FALSE)
```

5.2 Eigenvalues

```r
# a)
temp <- data.frame(pca$eig[,1])
temp %>%
  mutate(index = 1:10) %>%
  ggplot() +
  theme_bw() +
  geom_col(aes(x = index, y = pca.eig...1.)) +
  labs(
    title = "Eigenvalues",
    x = "index",
    y = "eigenvalues"
  )
```



b) The i^{th} eigenvalue (divided by total dimensions p) is interpreted as the percentage of variance captured by the dimension i.

c) The projection of originail data on the first dimension of the eigenvector explains 32.7 percent of total inertia. Those on the second dimension explains 17.4 percent of the inertia. Together, they contributes 50.1 percent of the explanation of total variance.

d) As a rule of thumb, we might arbitrarily choose 1 or 0.7 as a threshold for us to judge whether we retain a Principle Component. In both cases, we should choose to retain the top 4 dimensions with the highest eigenvalues. Collabratively they contributes to almost 75% of the total variation of the original data, and lowers the dimension as well.

5.3 Interpreting the PCs:

```
# eigenvectors (loadings)
pca$svd$V
```

```
##               [,1]       [,2]        [,3]        [,4]        [,5]
##  [1,] -0.42829627  0.1419891 -0.15557953 -0.03678703  0.36518741
##  [2,]  0.41015201 -0.2620794  0.15372674  0.09901016  0.04432336
##  [3,]  0.34414444  0.4539470 -0.01972378  0.18539458  0.13431954
##  [4,]  0.31619436  0.2657761 -0.21894349 -0.13189684  0.67121760
##  [5,] -0.37571570  0.4320460  0.11091758  0.02850297 -0.10597034
##  [6,] -0.41255442  0.1735910 -0.07815576  0.28290068  0.19857266
##  [7,]  0.30542571  0.4600244  0.03623770 -0.25259074 -0.12667770
##  [8,]  0.02783081 -0.1368411  0.58361717  0.53649480  0.39873734
##  [9,]  0.15319802  0.2405071 -0.32874217  0.69285498 -0.36873120
## [10,] -0.03210733  0.3598049  0.65987362 -0.15669648 -0.18557094
##               [,6]        [,7]        [,8]        [,9]       [,10]
##  [1,] -0.29607739  0.38177608  0.46160211  0.10475771  0.42428269
##  [2,]  0.30612478  0.62769317 -0.02101165  0.48266910  0.08104448
##  [3,] -0.30547299 -0.30972542 -0.31393005  0.42729075  0.39028424
##  [4,]  0.46777116 -0.09145002  0.12509166 -0.24366054 -0.10642724
##  [5,]  0.33252178 -0.12442114  0.21339819  0.55212939 -0.41399532
##  [6,]  0.09963776  0.35733030 -0.71111429 -0.15013429 -0.09086448
##  [7,] -0.44937288  0.42988982  0.03838986 -0.15480715 -0.44916580
##  [8,] -0.26166458 -0.09796019  0.17803824 -0.08297769 -0.27645138
##  [9,]  0.16320268  0.10674519  0.29614206 -0.24732691  0.08777340
## [10,]  0.29826888  0.08362898  0.01371744 -0.30773397  0.42923132
```

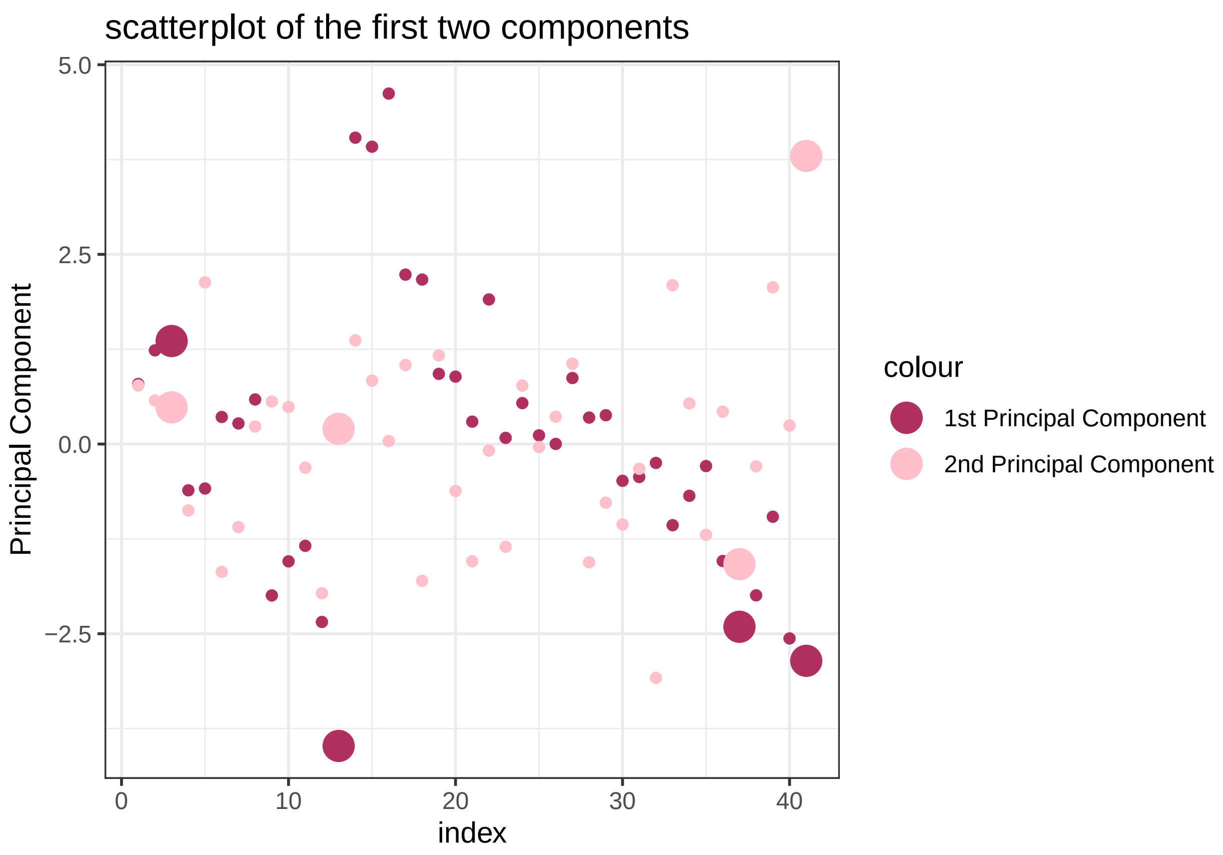
```
# correlations between variables and PCs
pca$var$cor
```

```
##                  Dim.1      Dim.2       Dim.3       Dim.4       Dim.5
## 100m        -0.77471983  0.1871420 -0.18440714 -0.03781826  0.30219639
## Long.jump    0.74189974 -0.3454213  0.18221105  0.10178564  0.03667805
## Shot.put     0.62250255  0.5983033 -0.02337844  0.19059161  0.11115082
## High.jump    0.57194530  0.3502936 -0.25951193 -0.13559420  0.55543957
## 400m        -0.67960994  0.5694378  0.13146970  0.02930198 -0.08769157
## 110m.hurdle -0.74624532  0.2287933 -0.09263738  0.29083103  0.16432095
## Discus       0.55246652  0.6063134  0.04295225 -0.25967143 -0.10482712
## Pole.vault   0.05034151 -0.1803569  0.69175665  0.55153397  0.32995932
## Javeline     0.27711085  0.3169891 -0.38965541  0.71227728 -0.30512892
## 1500m       -0.05807706  0.4742238  0.78214280 -0.16108904 -0.15356189
##                  Dim.6       Dim.7        Dim.8       Dim.9      Dim.10
## 100m        -0.22920075  0.25645445  0.290800753  0.04855323  0.18111827
## Long.jump    0.23697868  0.42164691 -0.013236949  0.22370807  0.03459636
## Shot.put    -0.23647411 -0.20805510 -0.197770097  0.19804125  0.16660497
## High.jump    0.36211310 -0.06143068  0.078805424 -0.11293209 -0.04543178
## 400m         0.25741324 -0.08357871  0.134436894  0.25590161 -0.17672678
## 110m.hurdle  0.07713202  0.24003322 -0.447988786 -0.06958442 -0.03878833
## Discus      -0.34787054  0.28877439  0.024184898 -0.07175021 -0.19174040
## Pole.vault  -0.20256095 -0.06580383  0.112160780 -0.03845860 -0.11801187
## Javeline     0.12633919  0.07170506  0.186563995 -0.11463138  0.03746881
## 1500m        0.23089724  0.05617697  0.008641731 -0.14262892  0.18323074
```

According to the loadings and correlation matrix, we can see that the athletes' performance on the first 7 events (from 100m to 110m hurdle) might be well explained by the first Princical Component, which a correlation mostly >.6 between the variables and the PCs. We might name it jump-advantage/running disadvantage, as the sign in the eigenvectors of running-scores are all negative. (Note that High jump might be an expection, if we arbitrarily set .6 as a threshold. However, the first principal component still explains more of the variation than any other components. So we leave it here.)
Simimarly, the score of Discus is best explained by the second dimension, Pole vault and 1500m by the third, and Javeline by the fourth.

5.4 Plot of Individuals:

```r
temp <- data.frame(pca$ind$coord[,1:2]) %>%
  mutate(name = rownames(pca$ind$coord[,1:2])) %>%
  mutate(index = 1:41)
temp2 <- temp %>%
  filter(name %in% c("KARPOV","BOURGUIGNON","Casarsa","Lorenzo"))
ggplot() +
  theme_bw() +
  geom_point(data = temp, aes(x = index, y = Dim.1, color = "1st Principal Component")) +
  geom_point(data = temp, aes(x = index, y = Dim.2, color = "2nd Principal Component")) +
  geom_point(data = temp2, aes(x = index, y = Dim.1, color = "1st Principal Component"), size = 5) +
  geom_point(data = temp2, aes(x = index, y = Dim.2, color = "2nd Principal Component"), size = 5) +
  labs(
    x = "index",
    y = "Principal Component",
    title = "scatterplot of the first two components"
  ) +
  scale_color_manual(values = c("maroon", "pink"))
```



Explanation: The graph above plots all the principal components for athletes in the events. Those expanded points denotes our focused athletes.
For Karpov in the decathlon, a high positive value of the first dimension indicates a better ability in jumping than running. It's not the case for Bourguignon, Lorenzo and Casarsa, where negative values of in first dimension suggest a strong running ability, but disadvantage in jumping.
As we discussed before, the second principal component simply denotes the ability of the event Discus. A positive correlation between the PC and the original data shows us that Casarsa performed excellently in this event, and Lorenzo is clearly not good at it. Karpov and Bourguignon are just a little better than average in this particular event.

Problem 6

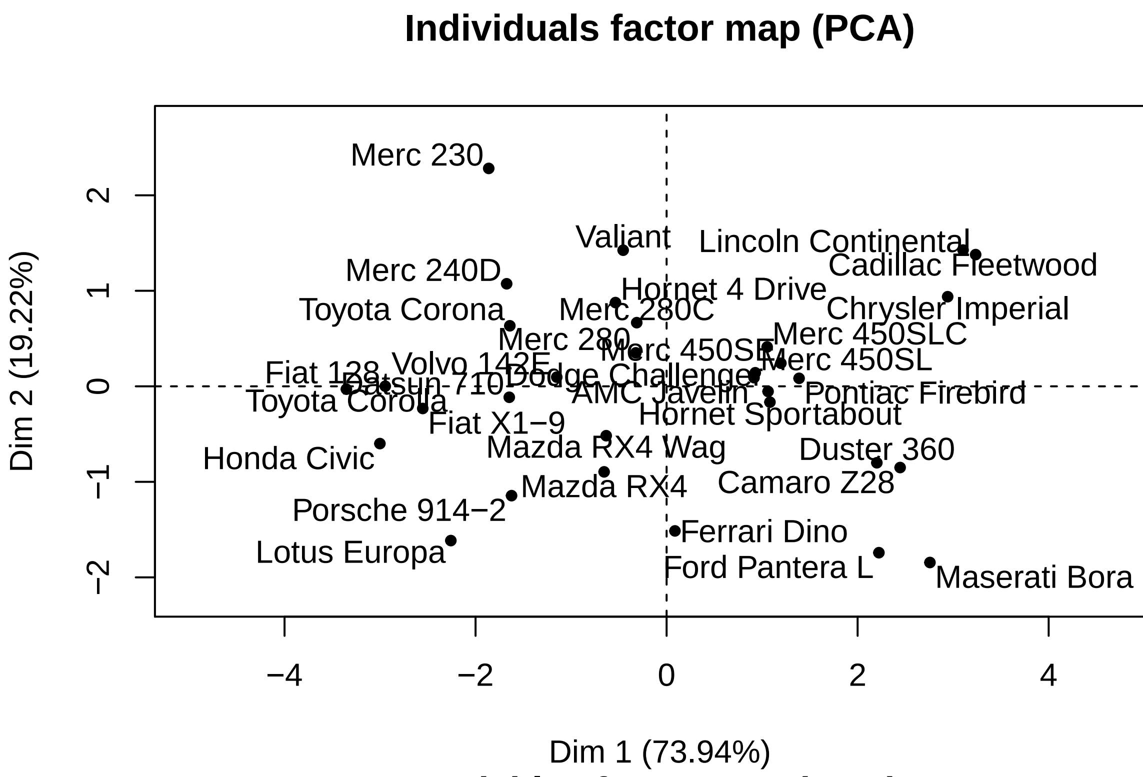
6.1 PCA-NIPALS algorithm

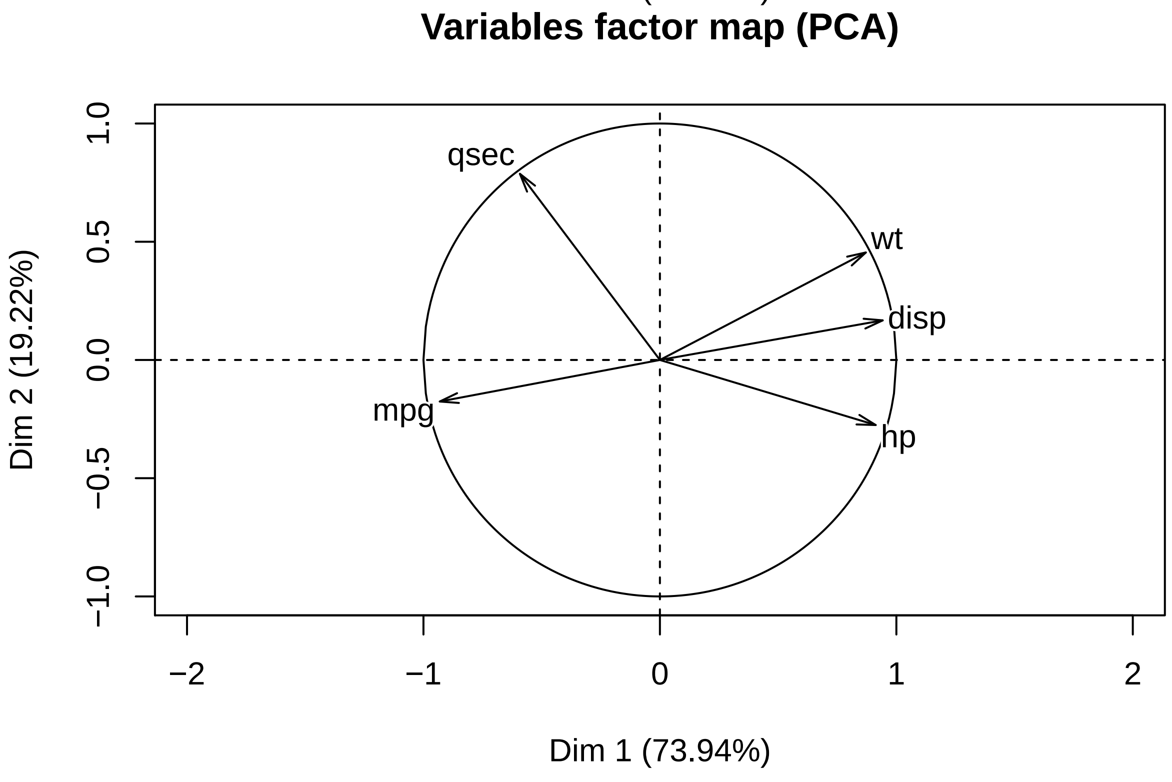
```r
# starting code
M <- as.matrix(mtcars[ ,c('mpg', 'disp', 'hp', 'wt', 'qsec')])
X <- scale(M)
```

```r
# An Overview of the Functions:
## check_convergence
### input: vector a, vector b
### output: boolean, True if a and b are close enough (i.e. converged), False if otherwise

## MyPca
### input: matrix X, with rows indicating the obs., and columns indicating the variables.
### output: environment, containing a matrix for eigenvectors "eigvec", eigenvalues "eigval", and PCs "
```

```r
check_convergence <- function(a, b){
  flag = TRUE
  for (i in 1:length(a)){
    if (abs(a[i] - b[i]) > 1e-5){
      flag = FALSE
    }
  }
  return(flag)
}
MyPca <- function(X){
  tempX = X
  eig <- environment()
  for (h in 1:ncol(X)) {
    w = rep(1,time = ncol(tempX))
    tempw = rep(-1, time = ncol(tempX))
    while (check_convergence(w, tempw) == FALSE) {
      tempw = w
      w = w / sqrt(sum(w^2))
      z = tempX %*% w / sum(w^2)
      w = t(tempX) %*% z / sum(z^2)
    }
    if (h == 1){
      W = w
      Z = z
    }else{
      W = W %>% append(w)
      Z = Z %>% append(z)
    }
    tempX = tempX - z %*% t(w)
  }
  W = matrix(W, ncol = ncol(tempX))
  Z = matrix(Z, ncol = ncol(tempX))
  assign("PC", Z, envir = eig)
  assign("eigvec", W, envir = eig)
  return(eig)
}
evd <- MyPca(X)
# Note that this should be roughly the same as the following results calculated by bulit-in functions
temp <- PCA(X, scale.unit = TRUE)
```





```
evd$eigvec
```

```
##             [,1]       [,2]       [,3]       [,4]        [,5]
## [1,] -0.4841976 -0.1791930 0.78015846  0.1333803  0.327111634
## [2,]  0.4900788  0.1708054 0.61194990 -0.2854847 -0.524084198
## [3,]  0.4746335 -0.2807383 0.08727133  0.8296312  0.002321402
## [4,]  0.4530419  0.4637240 0.09407156 -0.1142597  0.746864746
## [5,] -0.3079798  0.8030327 0.02010869  0.4465031 -0.246010965
```

```
temp$svd$V
```

```
##            [,1]       [,2]       [,3]       [,4]         [,5]
## [1,] -0.4841972 -0.1791977 0.78014898  0.1334381  0.327113277
## [2,]  0.4900784  0.1708092 0.61196922 -0.2854349 -0.524087714
## [3,]  0.4746341 -0.2807344 0.08721615  0.8296376  0.002331622
## [4,]  0.4530409  0.4637278 0.09408108 -0.1142560  0.746863338
## [5,] -0.3079815  0.8030300 0.02008331  0.4465067 -0.246005464
```

```
# Another way of confirmation is to verify X = Z %>% t(W) (top 6 rows displayed)
(X - evd$PC %*% t(evd$eigvec)) %>%
  head()
```

```
##                             mpg         disp            hp            wt
## Mazda RX4          0.000000e+00 0.000000e+00  0.000000e+00  0.000000e+00
## Mazda RX4 Wag      8.326673e-17 0.000000e+00 -1.110223e-16  5.551115e-17
## Datsun 710         0.000000e+00 0.000000e+00  0.000000e+00 -1.110223e-16
## Hornet 4 Drive     8.326673e-17 0.000000e+00  0.000000e+00 -1.600282e-16
## Hornet Sportabout  2.775558e-17 0.000000e+00  0.000000e+00 -8.326673e-17
## Valiant           -1.110223e-16 6.245005e-17  0.000000e+00  2.775558e-17
##                            qsec
## Mazda RX4          0.000000e+00
## Mazda RX4 Wag      1.110223e-16
## Datsun 710         5.551115e-17
## Hornet 4 Drive    -3.330669e-16
## Hornet Sportabout  1.110223e-16
## Valiant            0.000000e+00
```

```
# Hooray!
```

6.2 Calculate the eigenvalues by t(W)SW, where S is the covariance matrix

```
tempX <- scale(X)
t(evd$eigvec) %*% cov(tempX) %*% evd$eigvec
```

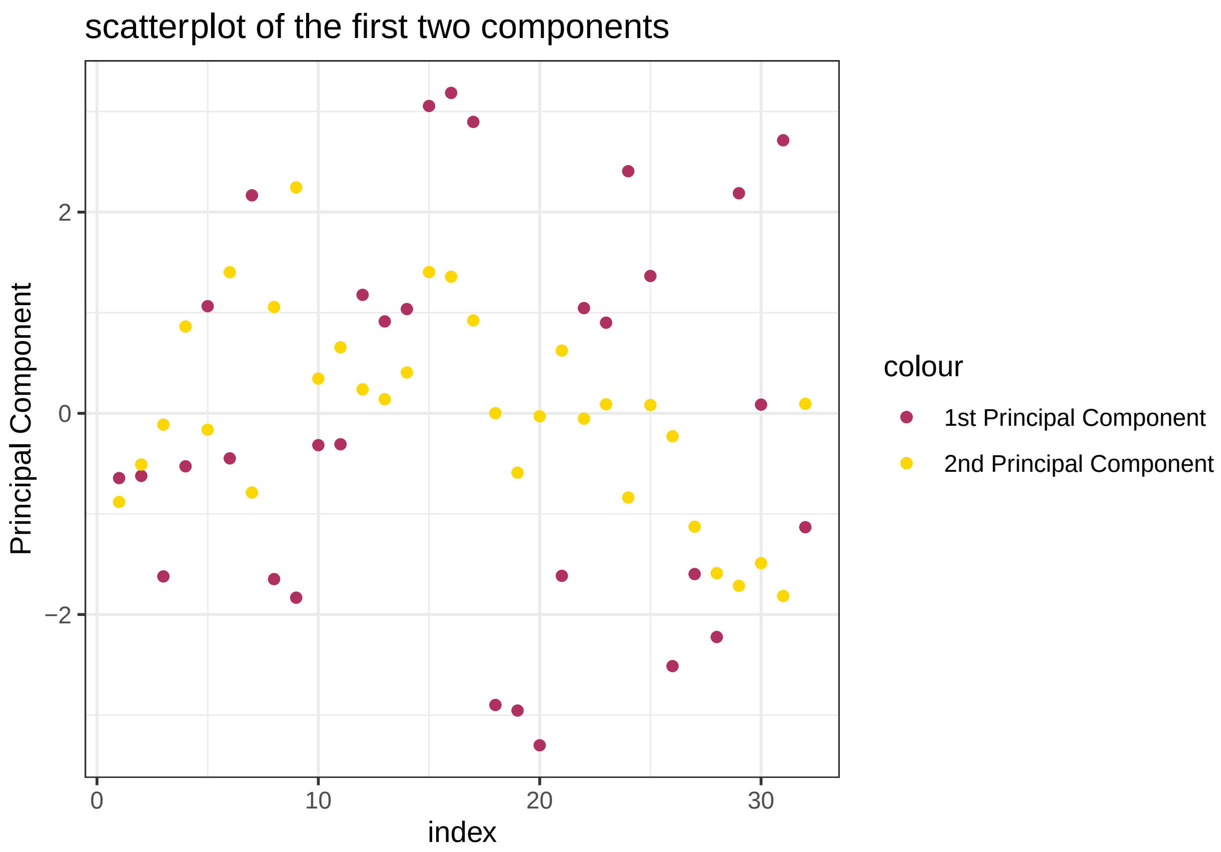
```
##               [,1]          [,2]          [,3]          [,4]          [,5]
## [1,]  3.696824e+00 -2.920508e-05  1.286616e-10  2.606071e-11  2.220446e-14
## [2,] -2.920508e-05  9.610168e-01 -4.241655e-06 -8.587352e-07 -7.153331e-10
## [3,]  1.286616e-10 -4.241655e-06  1.545313e-01 -3.308968e-06  4.076523e-11
## [4,]  2.606065e-11 -8.587352e-07 -3.308968e-06  1.311541e-01 -1.316107e-06
## [5,]  2.218364e-14 -7.153332e-10  4.076519e-11 -1.316107e-06  5.647363e-02
```

```
# Note that cov() can also be replaced as \frac{tempX^T tempX}{n}
# The results can also be confirmed by the built-in funciton.
temp$eig[,1]
```

```
##     comp 1     comp 2     comp 3     comp 4     comp 5
## 3.69682419 0.96101676 0.15453129 0.13115414 0.05647363
```

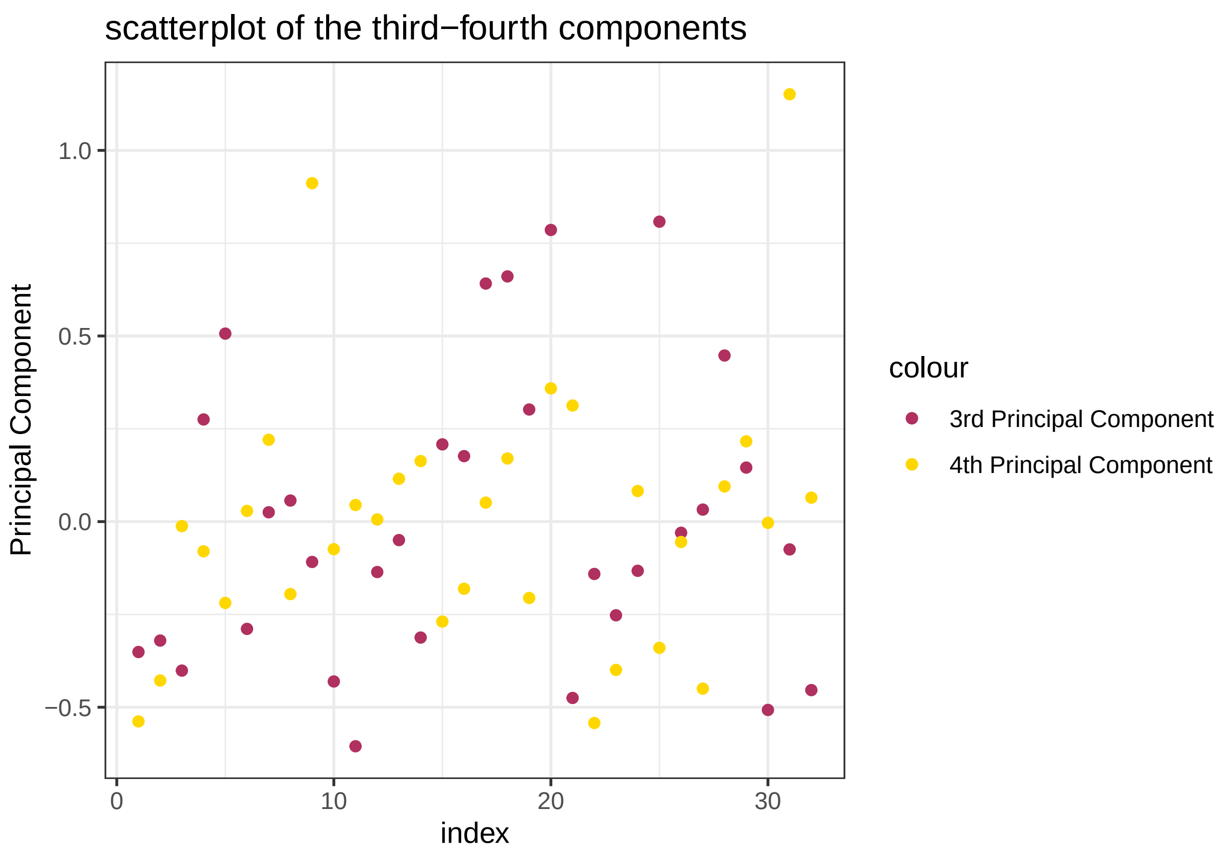
6.3 Two scatterplots of PCs
First 2 Dimensions:

```
temp2 <- data.frame(evd$PC[,1:2]) %>%
  mutate(index = 1:nrow(evd$PC[,1:2]))
temp2 %>%
ggplot() +
  theme_bw() +
  geom_point(aes(x = index, y = X1, color = "1st Principal Component")) +
  geom_point(aes(x = index, y = X2, color = "2nd Principal Component")) +
  labs(
    x = "index",
    y = "Principal Component",
    title = "scatterplot of the first two components"
  ) +
  scale_color_manual(values = c("maroon", "gold"))
```



Next 2 Dimensions:

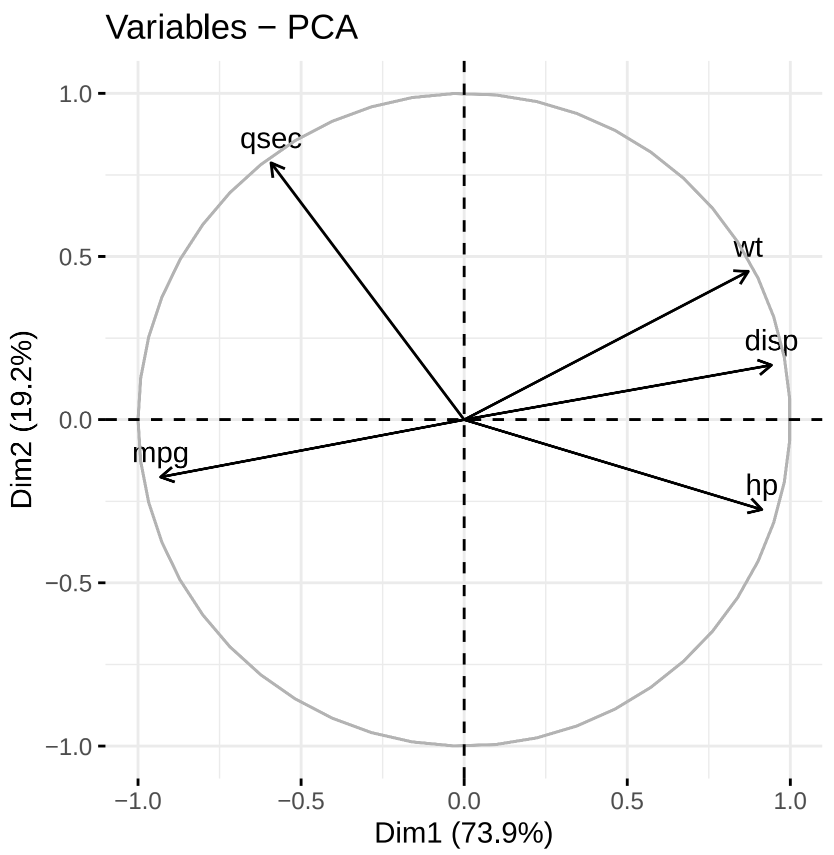
```
temp2 <- data.frame(evd$PC[,3:4]) %>%
  mutate(index = 1:nrow(evd$PC[,3:4]))
temp2 %>%
ggplot() +
  theme_bw() +
  geom_point(aes(x = index, y = X1, color = "3rd Principal Component")) +
  geom_point(aes(x = index, y = X2, color = "4th Principal Component")) +
  labs(
    x = "index",
    y = "Principal Component",
    title = "scatterplot of the third-fourth components"
  ) +
  scale_color_manual(values = c("maroon", "gold"))
```



Remark: The variation of observations in the first two dimensions (i.e. Principal Components) are much larger than that in the 3rd and 4th dimensions.

Plot a Circle of Correlations graph using the first tow dimensions (associated to the first 2 PCs).

```
fviz_pca_var(temp, col.var = "black")
```



The first two principal components are the projection of the orignal data on two orthonogal dimensions that retains the largest inertia. The first dimension in this case explains a huge part of variations in mpg, weight, displacement, and gross hp. It might be considered as an index of car-size. The second principal component, on the other hand, provides an indicaiton of the speed of the car. Together, the two PCs contributes 92.1% of the total variations of the original car-properties data.