hw2

caojilin

2/12/2019

Honor Code

The honor code

(a) Please state the names of people who you worked with for this homework. You can also provide your comments about the homework here.

> Cinidy Liu

(b) Please type/write the following sentences yourself and sign at the end. We want to make it *extra* clear that nobody cheats even unintentionally.

I hereby state that all of my solutions were entirely in my words and were written by me. I have not looked at another student's solutions and I have fairly credited all external sources in this write up.

> I hereby state that all of my solutions were entirely in my words and were written by me. I have not looked at another student's solutions and I have fairly credited all external sources in this write up.

2

1.1

$$M = \begin{pmatrix} u_1 \cdots u_n \end{pmatrix} \begin{pmatrix} d_1 & & \\ & \ddots & \\ & & d_n \end{pmatrix} \begin{pmatrix} v_1^T \\ \vdots \\ v_n^T \end{pmatrix} = \sum_{i=1}^{n} u_i d_i v_i^T = \sum_{i=1}^{n} d_i u_i v_i^T$$

1.2

$$M^T M = VDU^T UDV^T = VD^2V^T = VD^2V^{-1} = \begin{pmatrix} v_1 \cdots v_n \end{pmatrix} \begin{pmatrix} d_1^2 & & \\ & \ddots & \\ & & d_n^2 \end{pmatrix} V^{-1}$$

eigenvalue d_i^2 corresponding to eigenvector v_i

$$MM^T = UDV^T VDU^T = UD^2U^{-1} = \begin{pmatrix} u_1 \cdots u_n \end{pmatrix} \begin{pmatrix} d_i^2 & & \\ & \ddots & \\ & & d_n^2 \end{pmatrix} U^{-1}$$

eigenvalue d_i^2 corresponding to eigenvector u_i

1

1.3

```
n = c(2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048)
time_for_create = rep(0,length(n))
time_for_svd = rep(0,length(n))

for (i in 1:length(n)) {
  start =Sys.time()
  M = matrix(rnorm(n[i]*n[i]),n[i],n[i])
  end = Sys.time()
  time_for_create[i] = end - start

  start =Sys.time()
  svd(M)
  end = Sys.time()
  time_for_svd[i] = end - start
}
```

```
plot(n, log(time_for_create+1), type="l",ylab="time", col="red")
lines(n, log(time_for_svd+1), type="l", col="green")
legend('topright',c("time_for_create","time_for_svd") ,
   lty=1, col=c('red', 'green'), bty='n', cex=.75)
```

2.1 First eigenvector and eigenvector

```r
#a,b,c,e
power_method = function(A, d, k){
  #Start with an arbitrary vector w0
  w = rnorm(d)
  for (i in 1:k) {
    s = max(abs(A %*% w))
    w = (A %*% w) / s
  }
  w = w/ sqrt(sum(w^2))
  return(data.frame(w, s))
}
A  = matrix(c(1,2,3,2,-1,4,3,4,-5), 3,3,byrow = TRUE)
output = power_method(A,3,10)
w = output$w
s = output$s
```

```r
#d
(t(w) %*% A %*% w) / t(w) %*% w
```

```
##             [,1]
## [1,] -7.763459
```

```r
max(abs(eigen(A)$values))
```

```
## [1] 7.768189
```

```r
#they are very close
```

```r
eigen(A)
```

```
## eigen() decomposition
## $values
## [1]  4.610843 -1.842654 -7.768189
##
## $vectors
##            [,1]       [,2]       [,3]
## [1,] -0.6890036  0.6985555 -0.1931172
## [2,] -0.5672220 -0.6856083 -0.4562899
## [3,] -0.4511466 -0.2048451  0.8686226
```

```r
#
paste(c("eigenvector by power method: ", round(w,7)), collapse=" ")
```

```
## [1] "eigenvector by power method:  -0.2065485 -0.4672882 0.8596391"
```

```r
paste(c("eigenvalue by power method: ", s[1]), collapse=" ")
```

```
## [1] "eigenvalue by power method:  7.55998063767506"
```

2.2 Defation and more eigenvectors

```r
#a
B = matrix(c(5,1,0,1,4,0,0,0,1), 3, 3, byrow = TRUE)

output = power_method(B,3,20)
lambda1 = output$s
v1 = output$w
```

```r
#b
B1 = B - lambda1 * v1 %*% t(v1)
output2 = power_method(B1, 3, 20)
lambda2 = output2$s
v2 = output2$w
```

```r
#c
B2 = B1 - lambda2 * v2 %*% t(v2)
output3 = power_method(B2, 3, 20)
lambda3 = output3$s
v3 = output3$w
evalues = round(c(lambda1[1],lambda2[1],lambda3[1]),5);evalues
```

```
## [1] 5.61809 3.38197 1.00000
```

```r
evectors = matrix(round(c(v1,v2,v3),5),3,3);evectors
```

```
##          [,1]     [,2] [,3]
## [1,] -0.85064 -0.52577    0
## [2,] -0.52575  0.85063    0
## [3,]  0.00000  0.00000   -1
```

```r
eigen(B)
```

```
## eigen() decomposition
## $values
## [1] 5.618034 3.381966 1.000000
##
## $vectors
##           [,1]       [,2] [,3]
## [1,] 0.8506508  0.5257311    0
## [2,] 0.5257311 -0.8506508    0
## [3,] 0.0000000  0.0000000    1
```

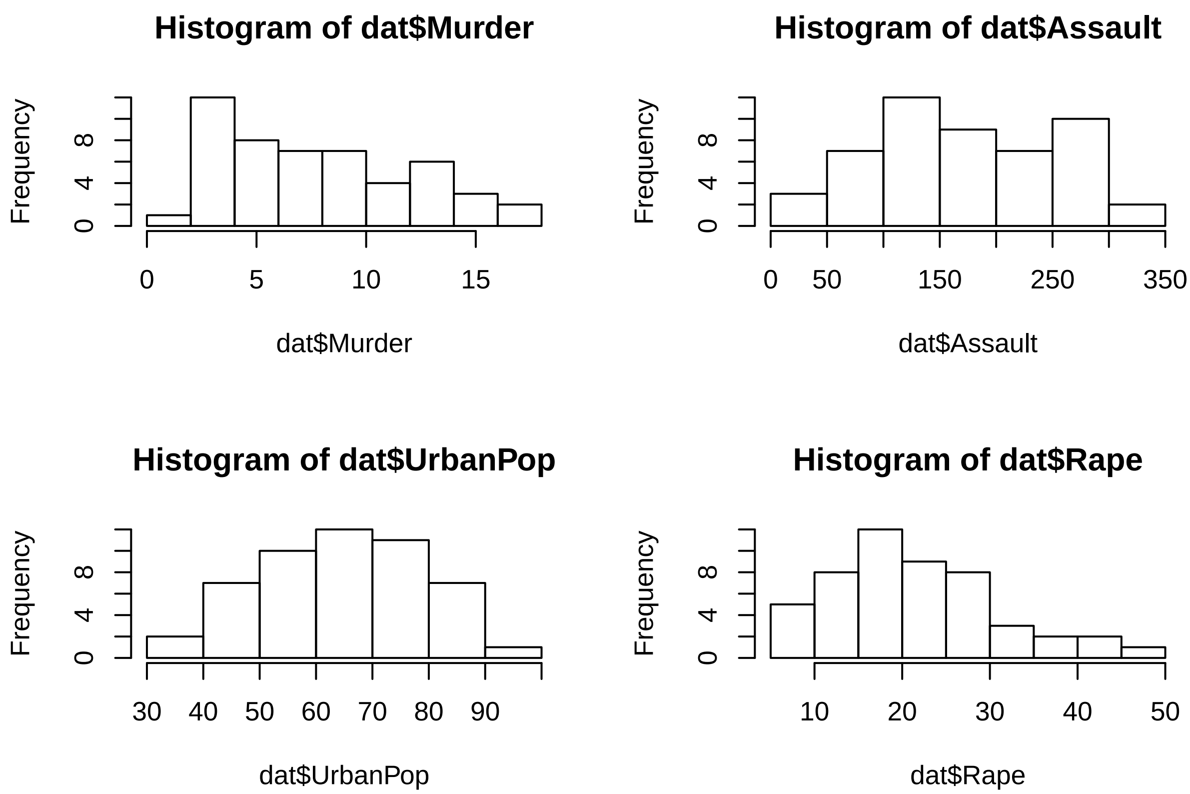
```r
#They are very close to R's output
```

3 Principal Component Analysis

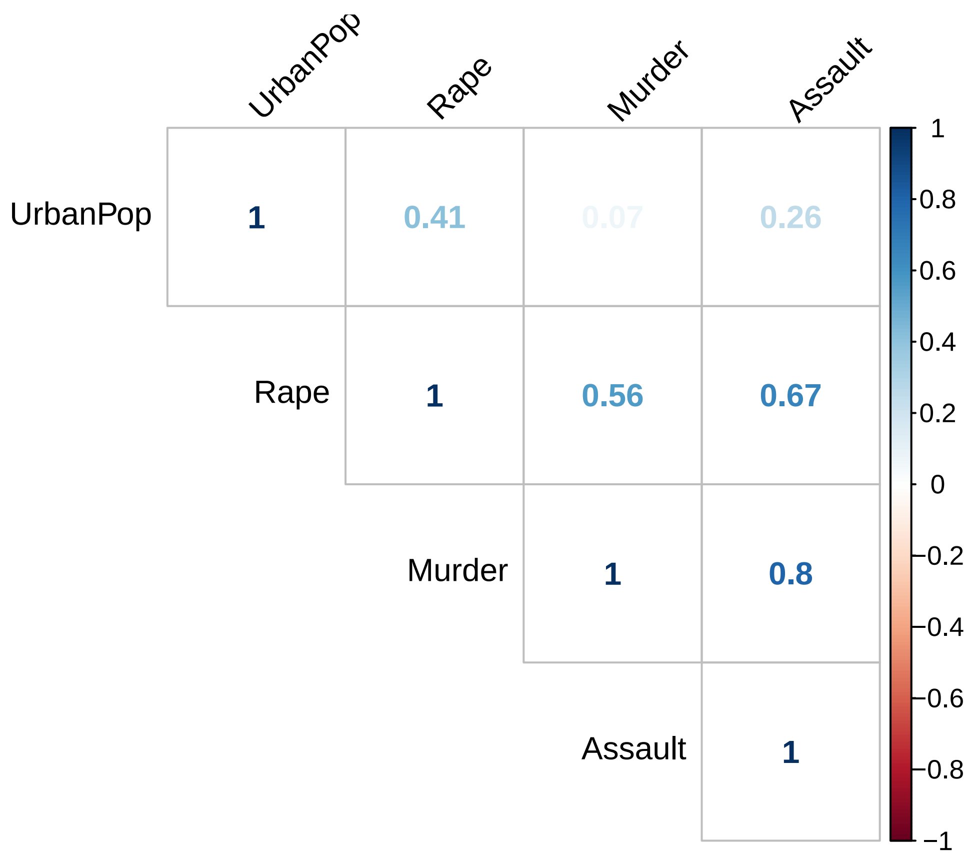
(a)

```
## [1] "mean by col"

##   Murder  Assault UrbanPop     Rape
##    7.788  170.760   65.540   21.232

## [1] "variance by col"

##     Murder    Assault   UrbanPop       Rape
##   18.97047 6945.16571  209.51878   87.72916
```

(b)



(c)



we see that crimes(Rape, Murder and Assualt) have high correlations to each other while UrbanPop does not.

(d)

```
## Importance of components:
##                           PC1    PC2     PC3     PC4
## Standard deviation     1.5749 0.9949 0.59713 0.41645
## Proportion of Variance 0.6201 0.2474 0.08914 0.04336
## Cumulative Proportion  0.6201 0.8675 0.95664 1.00000
```

(e)

```
#(e)Obtain the principal vectors and store them in a matrix, include row and column names.
#Display the first three loadings.
pc_vectors = pca_prcomp$rotation
#first three loadings
pca_prcomp$rotation[,1:3]
```

```
##                 PC1        PC2        PC3
## Murder   -0.5358995  0.4181809 -0.3412327
## Assault  -0.5831836  0.1879856 -0.2681484
## UrbanPop -0.2781909 -0.8728062 -0.3780158
## Rape     -0.5434321 -0.1673186  0.8177779
```

(f)

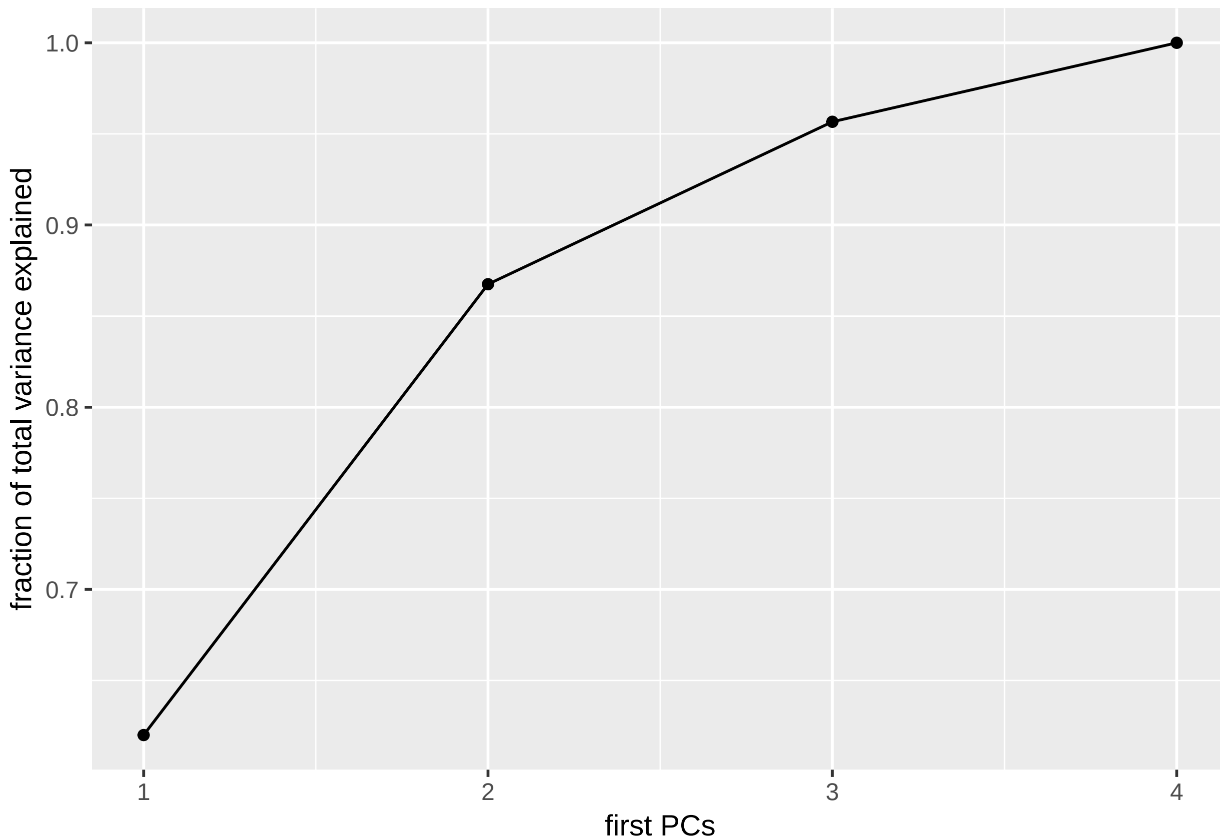
```
##                       PC1         PC2         PC3
## Alabama        -0.97566045  1.12200121 -0.43980366
## Alaska         -1.93053788  1.06242692  2.01950027
## Arizona        -1.74544285 -0.73845954  0.05423025
## Arkansas        0.13999894  1.10854226  0.11342217
## California     -2.49861285 -1.52742672  0.59254100
## Colorado       -1.49934074 -0.97762966  1.08400162
## Connecticut     1.34499236 -1.07798362 -0.63679250
## Delaware       -0.04722981 -0.32208890 -0.71141032
## Florida        -2.98275967  0.03883425 -0.57103206
## Georgia        -1.62280742  1.26608838 -0.33901818
## Hawaii          0.90348448 -1.55467609  0.05027151
## Idaho           1.62331903  0.20885253  0.25719021
## Illinois       -1.36505197 -0.67498834 -0.67068647
## Indiana         0.50038122 -0.15003926  0.22576277
## Iowa            2.23099579 -0.10300828  0.16291036
## Kansas          0.78887206 -0.26744941  0.02529648
## Kentucky        0.74331256  0.94880748 -0.02808429
## Louisiana      -1.54909076  0.86230011 -0.77560598
## Maine           2.37274014  0.37260865 -0.06502225
## Maryland       -1.74564663  0.42335704 -0.15566968
## Massachusetts   0.48128007 -1.45967706 -0.60337172
## Michigan       -2.08725025 -0.15383500  0.38100046
## Minnesota       1.67566951 -0.62590670  0.15153200
## Mississippi    -0.98647919  2.36973712 -0.73336290
## Missouri       -0.68978426 -0.26070794  0.37365033
## Montana         1.17353751  0.53147851  0.24440796
## Nebraska        1.25291625 -0.19200440  0.17380930
## Nevada         -2.84550542 -0.76780502  1.15168793
## New Hampshire   2.35995585 -0.01790055  0.03648498
## New Jersey     -0.17974128 -1.43493745 -0.75677041
## New Mexico     -1.96012351  0.14141308  0.18184598
## New York       -1.66566662 -0.81491072 -0.63661186
## North Carolina -1.11208808  2.20561081 -0.85489245
## North Dakota    2.96215223  0.59309738  0.29824930
## Ohio            0.22369436 -0.73477837 -0.03082616
## Oklahoma        0.30864928 -0.28496113 -0.01515592
## Oregon         -0.05852787 -0.53596999  0.93038718
## Pennsylvania    0.87948680 -0.56536050 -0.39660218
## Rhode Island    0.85509072 -1.47698328 -1.35617705
## South Carolina -1.30744986  1.91397297 -0.29751723
## South Dakota    1.96779669  0.81506822  0.38538073
## Tennessee      -0.98969377  0.85160534  0.18619262
## Texas          -1.34151838 -0.40833518 -0.48712332
## Utah            0.54503180 -1.45671524  0.29077592
## Vermont         2.77325613  1.38819435  0.83280797
## Virginia        0.09536670  0.19772785  0.01159482
## Washington      0.21472339 -0.96037394  0.61859067
## West Virginia   2.08739306  1.41052627  0.10372163
## Wisconsin       2.05881199 -0.60512507 -0.13746933
## Wyoming         0.62310061  0.31778662 -0.23824049
```

(g)

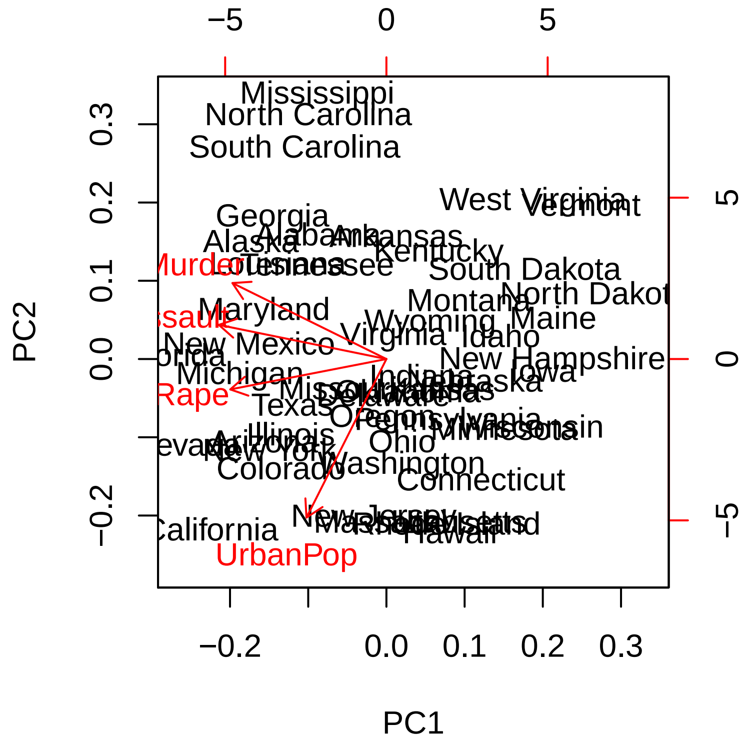
```
## [1] 2.4802416 0.9897652 0.3565632 0.1734301
```

```
## [1] 4
```

(h)

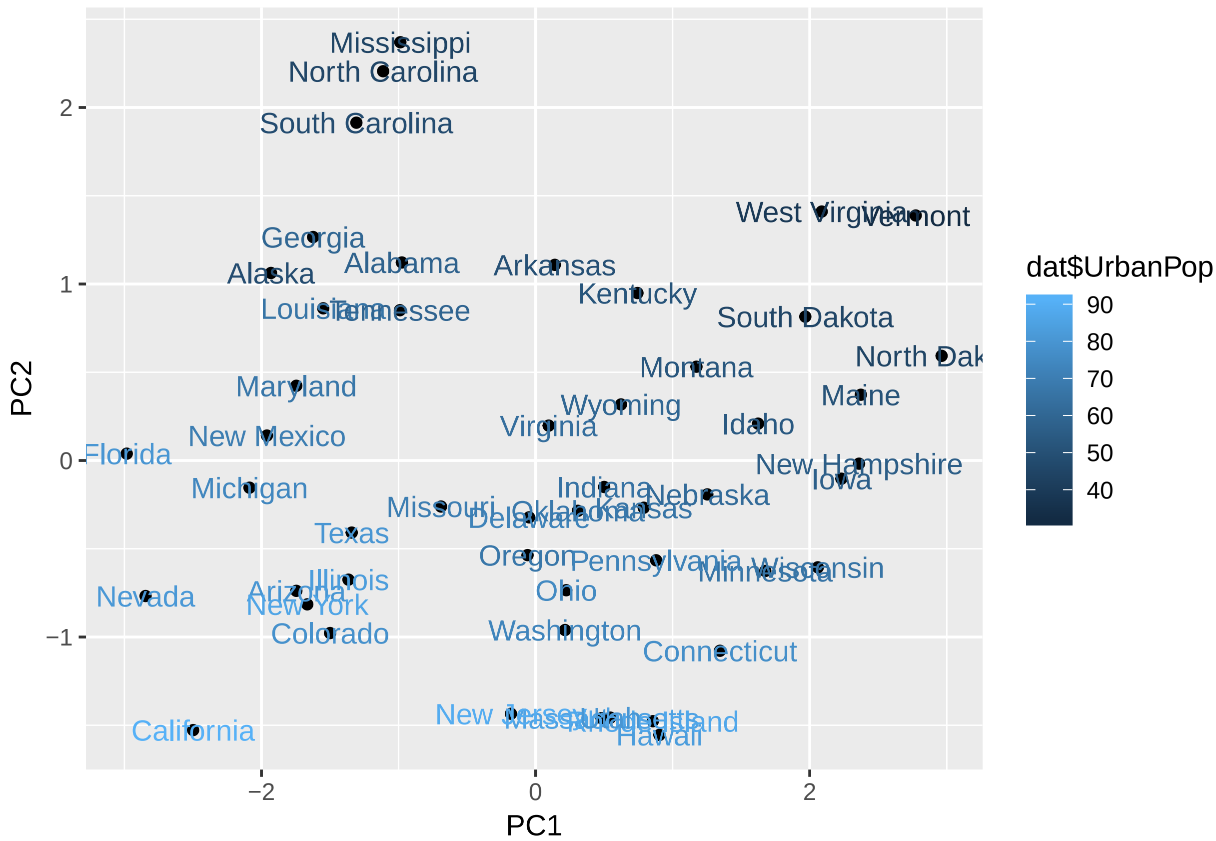


(i)

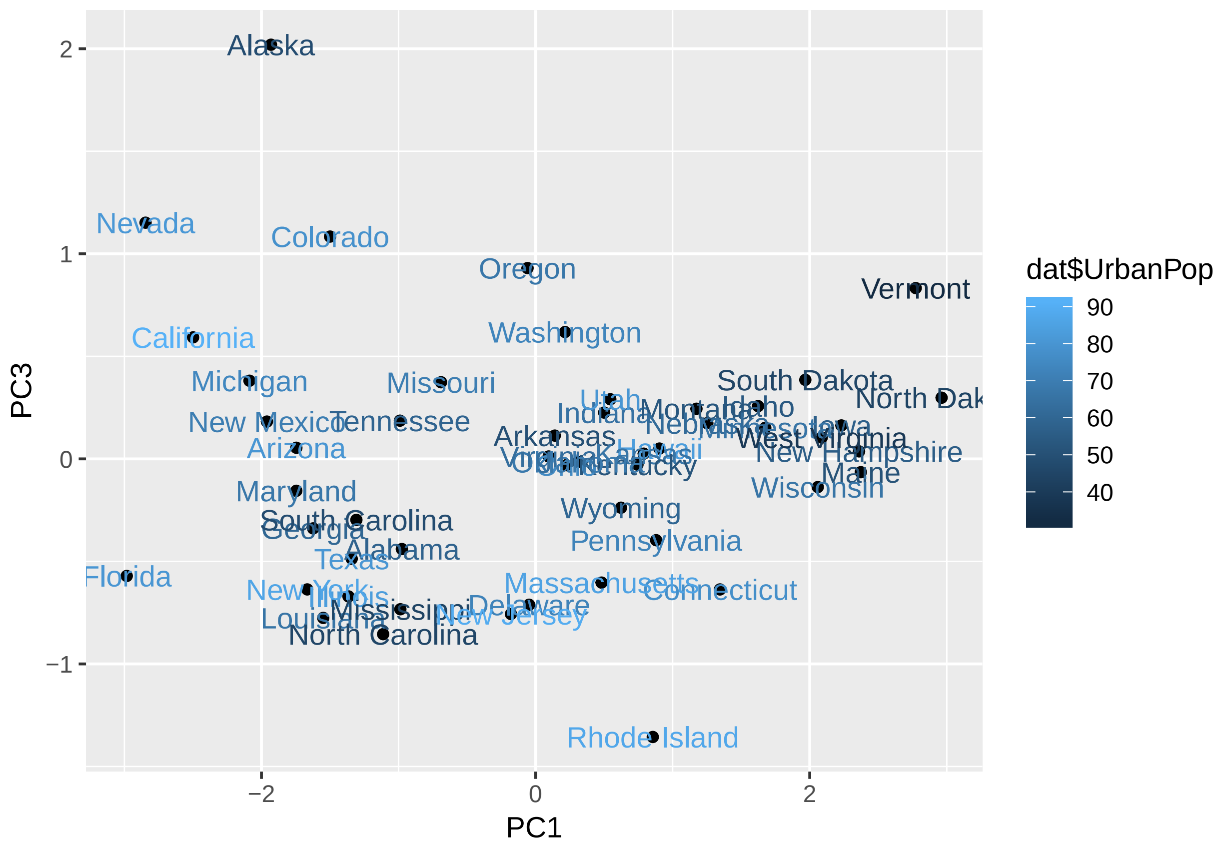


PC1 put equal weights on crimes and less weight on UrbanPop, so it measures overall crime rates
PC2 put more weight on UrbanPop, so it indicates the population level.
For example, California, Florida, Neveda have high population and high crime rates.
States like New Mexico, Michigan, Maryland have low population but high crime rates.

(j)


PC1
PC2
dat$UrbanPop
90
80
70
60
50
40
Mississippi
North Carolina
South Carolina
West Virginia
Vermont
Georgia
Alaska
Alabama
Arkansas
Kentucky
Louisiana
Tennessee
South Dakota
North Dak
Montana
Maine
Maryland
Wyoming
Virginia
Idaho
New Mexico
Florida
New Hampshire
Iowa
Michigan
Indiana
Nebraska
Missouri
Oklahoma
Kansas
Delaware
Texas
Oregon
Pennsylvania
Minnesota
Wisconsin
Nevada
Arizona
Illinois
New York
Ohio
Colorado
Washington
Connecticut
New Jersey
Massachusetts
Rhode Island
Utah
Hawaii
California

(k)



PC1 put equal weights on crimes and less weight on UrbanPop, so it measures overall crime rates
PC3 has more weight on Rape, so it measures the Rape rates.
So this plot displays relationship between overrall crimes and Rape.
Nevada, Alaska and Colorado stand out for high Rape rates.
Rhode Island has high UrbanPop but low Rape rates and crime rates.

4 K-means and PCA

1. This problem involves the K-means clustering algorithm.

(a) Prove (10.12).

$$\frac{1}{|C_k|}\sum_{i,i'\in C_k}\sum_{j=1}^{p}(x_{ij}-x_{i'j})^2 = 2\sum_{i\in C_k}\sum_{j=1}^{p}(x_{ij}-\bar{x}_{kj})^2 = \frac{1}{|C_k|}\sum_{i,i'\in C_k}\sum_{j=1}^{p}((x_{ij}-\bar{x}_{kj})-(x_{i'j}-\bar{x}_{kj}))^2 = \frac{1}{|C_k|}\sum_{i,i'\in C_k}\sum_{j=1}^{p}((x_{ij}-\bar{x}_{kj})^2-2($$

(b) On the basis of this identity, argue that the K-means clustering algorithm (Algorithm 10.1) decreases the objective (10.11) at each iteration.

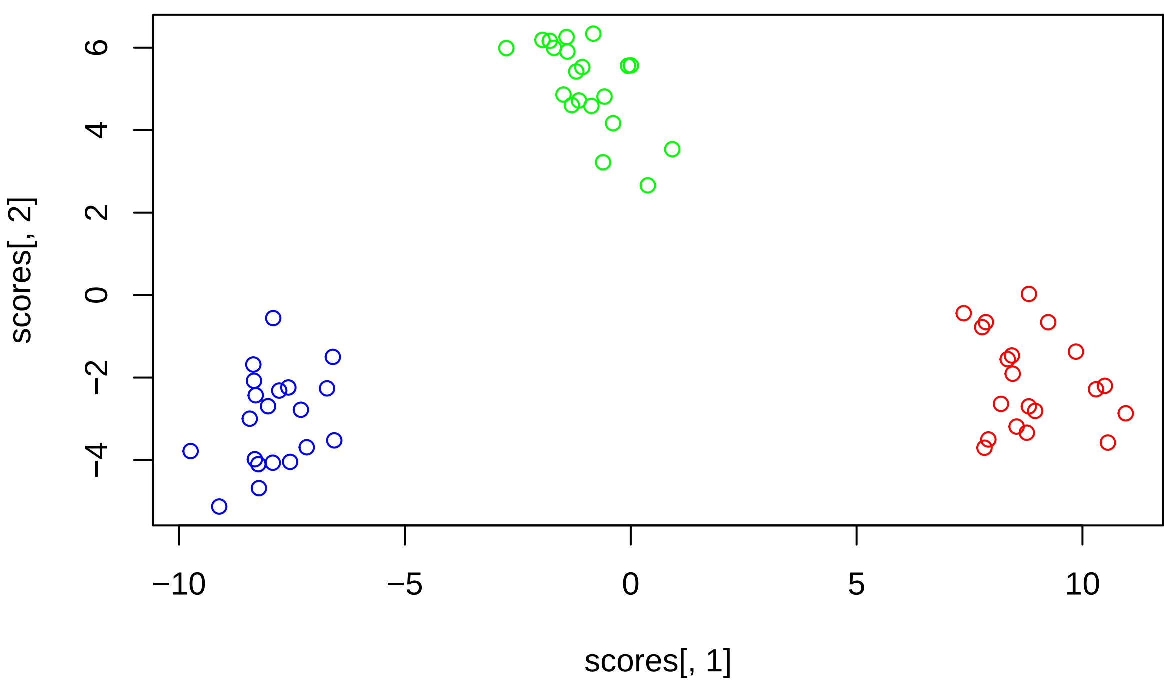
above equation shows that K-means clustering algorithm equivalently decreases the within-cluster variation.
since we recompute the centroid using squared Euclidean distance
Step 2(b) will minimize the within-cluster sum of squares.
Therefore it will decreases the objective each iteration.

10. In this problem, you will generate simulated data, and then perform PCA and K-means clustering on the data.

(a)

Generate a simulated data set with 20 observations in each of three classes (i.e. 60 observations total), and 50 variables.

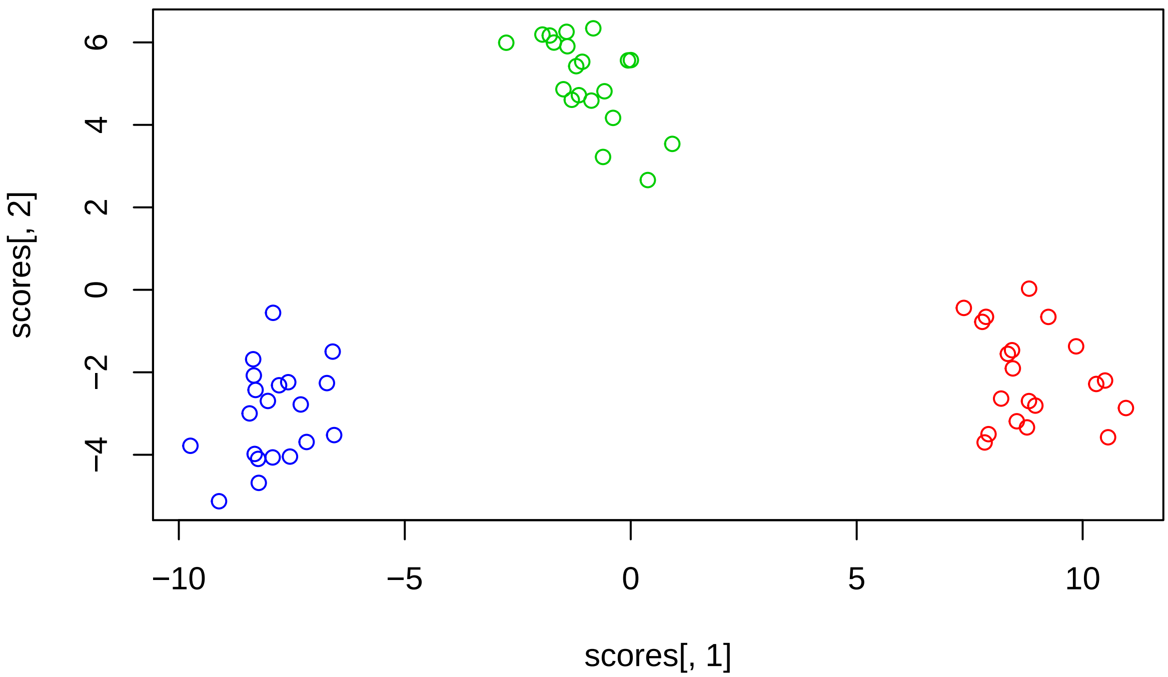
(b)



(c)

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [36] 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
##
##       1  2  3
##    1 20  0  0
##    2  0  0 20
##    3  0 20  0
```

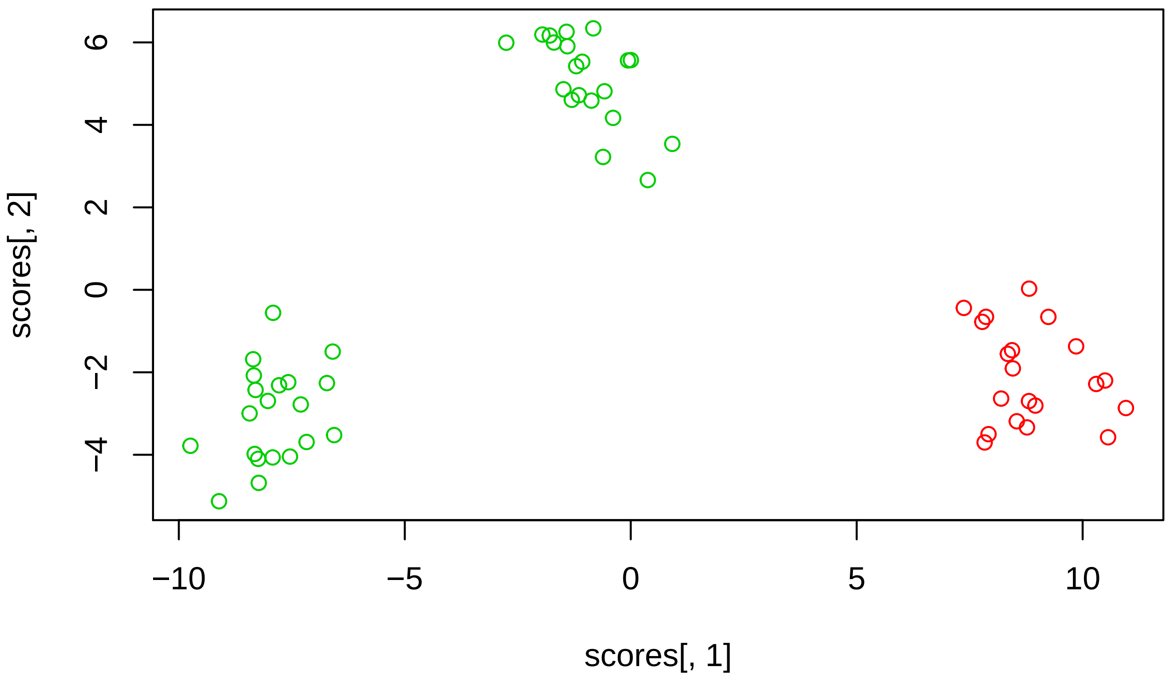


When $K = 3$, we see that the k-mean successfully classify the data

(d)

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [36] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
##
##       1  2  3
##    1 20  0  0
##    2  0 20 20
```

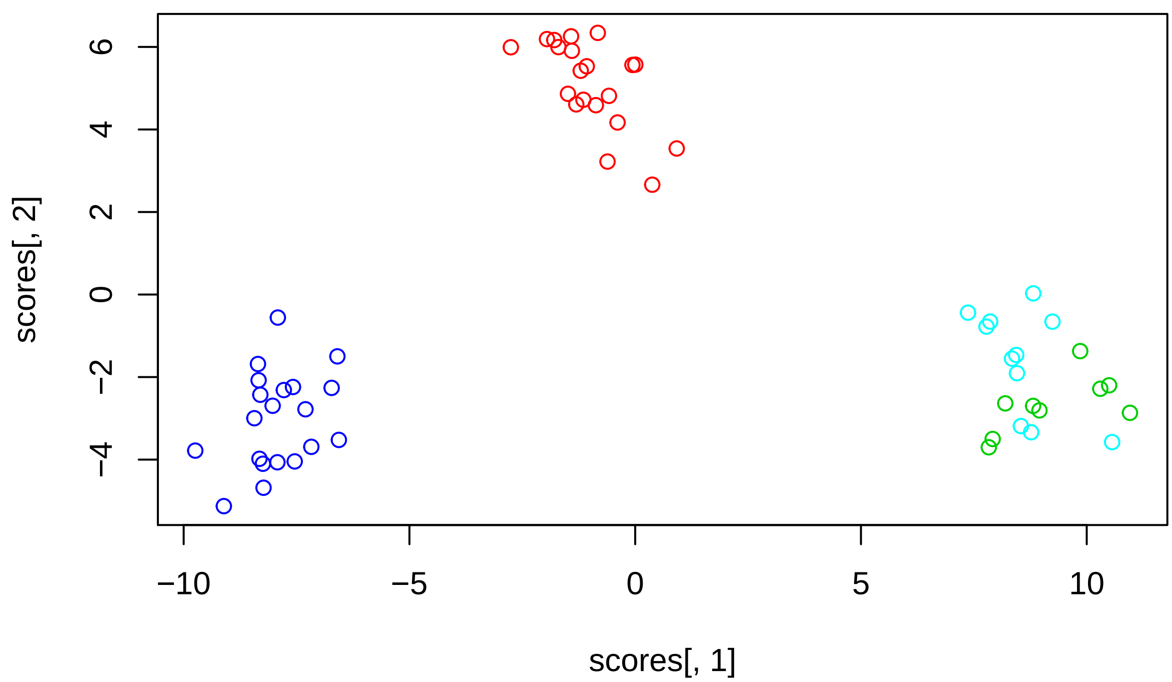


When $K = 2$, the second cluster obtained contains the observations of both first the second cluster based on original true labels.

(e)

```
##  [1] 4 4 4 4 4 2 4 2 4 4 2 2 2 2 4 2 4 2 2 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [36] 3 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
##
##     TRUE1 TRUE2 TRUE3
##   1     0     0    20
##   2     9     0     0
##   3     0    20     0
##   4    11     0     0
```



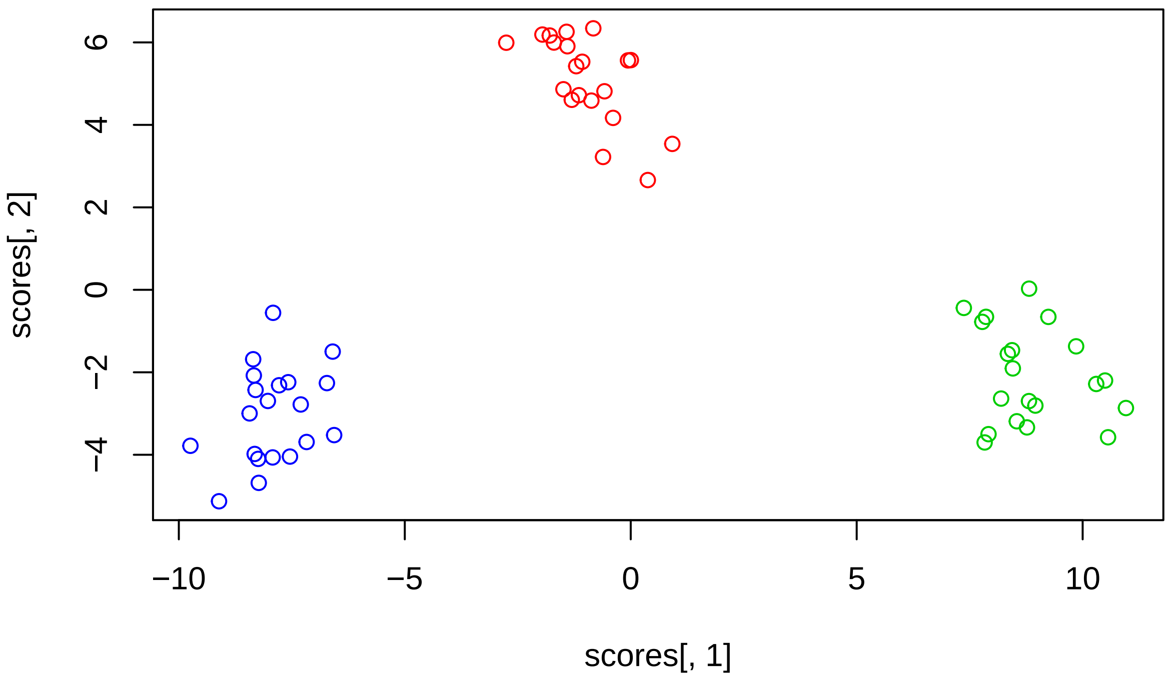
When $K = 4$, the first and thrid clusters obtained from the observations of third cluster based on original true labels.

(f)

perform K-means clustering with K = 3 on the first two
principal component score vectors

```
##  [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [36] 3 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

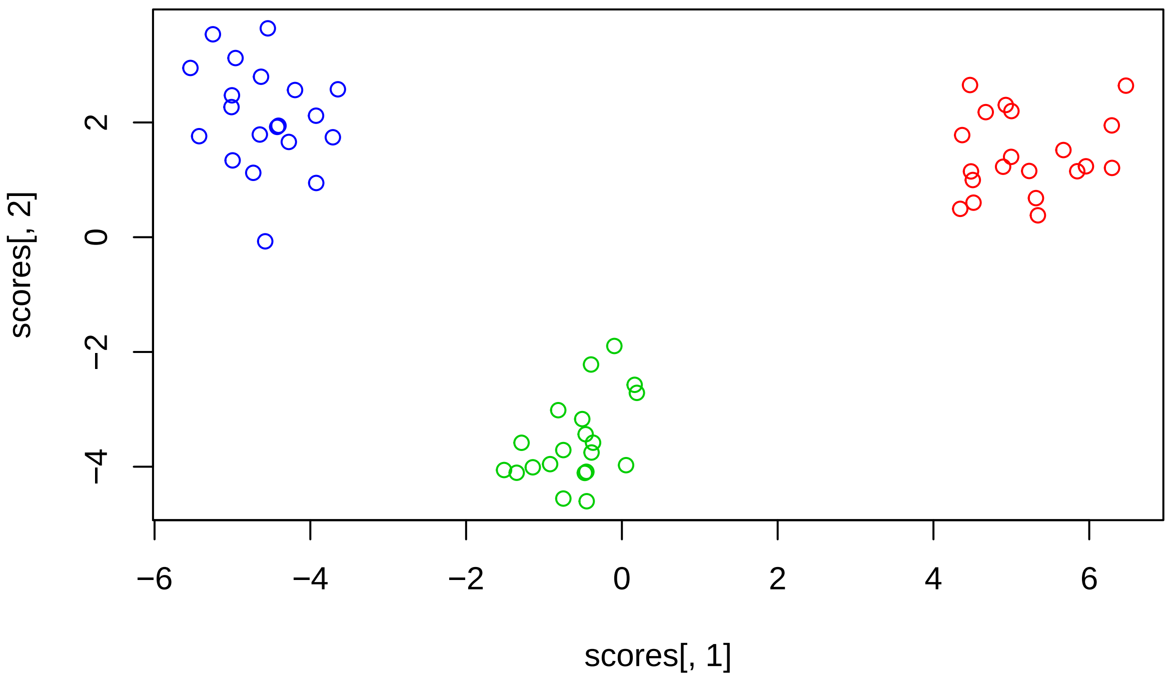
```
##
##     TRUE1 TRUE2 TRUE3
##   1     0     0    20
##   2    20     0     0
##   3     0    20     0
```



(g)

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [36] 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
##
##     TRUE1 TRUE2 TRUE3
##   1    20     0     0
##   2     0     0    20
##   3     0    20     0
```



After scaling, it still gives the same result as in part b

5 True or false

Examine whether the following statements are true or false and provide one line justification

Eigenvalues obtained from the principal component analysis is always nonnegative.
True, $X^T X$ is a symmetric and PSD matrix, thus it has positive eigenvalues

The first principal vector and the second principal vector are always orthogonal.
True, actually all principal vectors are othrogonal to each other

Singular values of a square matrix M are the same as the eigenvalues of M.
False, when the matrix M is a symmetric positive definite matrix.

Principal components analysis can be used to create a low dimensional projection of the data.
True, use principal vectors as new coordinates

Eigenvalue of a matrix are always nonnegative.
False, Not always. Eigenvalues can be negative.

The y-axis of a Scree plot is always from 0 to 1.
True, We scales it to 0 to 1 so that it's convenient for us to see proportion

The maximum number of principal components is always less or equal to the feature dimension.
True, When n > p, maximum number of principal components are equal to p, when n < p, maximum number of principal components are equal to n.