Principal Component Analysis

MSBA7002: Business Statistics

Contents

Contents

PCA: Principal Component Analysis

Chapter 6.3 and Chapter 12.1-12.3.

0. Dimension reduction

- capture the main features
- cut the noise hidden in the data
- visualization of large dimension

1. PC's interpretations

- The best low dimension of linear approximation to the data (or closest to the data)
- The direction of linear combination which has largest variance
- We may take a small number of PC's as a set of input… to other analyses

Part I: Case study: Measurement of Intelligence from ASVAB tests

Goal: How people differ in intelligence?

- Our data set IQ.csv is a subset of individuals from the 1979 National Longitudinal Study of Youth (NLSY79) survey who were re-interviewed in 2006.
- Information about family, personal demographic such as gender, race and eduction level, plus a set of ASVAB (Armed Services Vocational Aptitude Battery) test scores.
- It is STILL used as a screening test for those who want to join the US army!

The test has the following components:

- Science, Arith (Arithmetic reasoning), Word (Word knowledge), Parag (Paragraph comprehension), Numer (Numerical operation), Coding (Coding speed), Auto (Automative and Shop information), Math (Math knowledge), Mechanic (Mechanic Comprehension) and Elec (Electronic information).
- Lastly AFQT (Armed Forces Qualifying Test) is a combination of Word, Parag, Math and Arith.
- Note: Service Branch requirement: Army 31, Navy 35, Marines 31, Air Force 36, and Coast Guard 45, (out of 100 which is the max!) My prediction is that all of us pass the requirements, even for Coast Guard. :)

Our goal is to see how we can summarize the set of tests and grab main information about each one's intelligence efficiently.

Note: One of the original study goals is to see how intelligence relates to one's future successes measured by income in 2005.

0) Get a quick look at the data

```
data1 <- read.csv("IQ.Full.csv")
#dim(data1)
names(data1)
```

```
##  [1] "Subject"        "Imagazine"      "Inewspaper"     "Ilibrary"
##  [5] "MotherEd"       "FatherEd"       "FamilyIncome78" "Race"
##  [9] "Gender"         "Educ"           "Science"        "Arith"
## [13] "Word"           "Parag"          "Numer"          "Coding"
## [17] "Auto"           "Math"           "Mechanic"       "Elec"
## [21] "AFQT"           "Income2005"     "Esteem1"        "Esteem2"
## [25] "Esteem3"        "Esteem4"        "Esteem5"        "Esteem6"
## [29] "Esteem7"        "Esteem8"        "Esteem9"        "Esteem10"
```

```
#summary(data1)
```

1) We first concentrate on the AFQT tests: Word, Math, Parag and Arith

Question:

i) How can we best capture the performance based on the four tests?

ii) Can we come up with some sensible scores combining the four tests together?

Note:

ii) is similar to the creation of SP500, a weighted index based on 500 stocks.

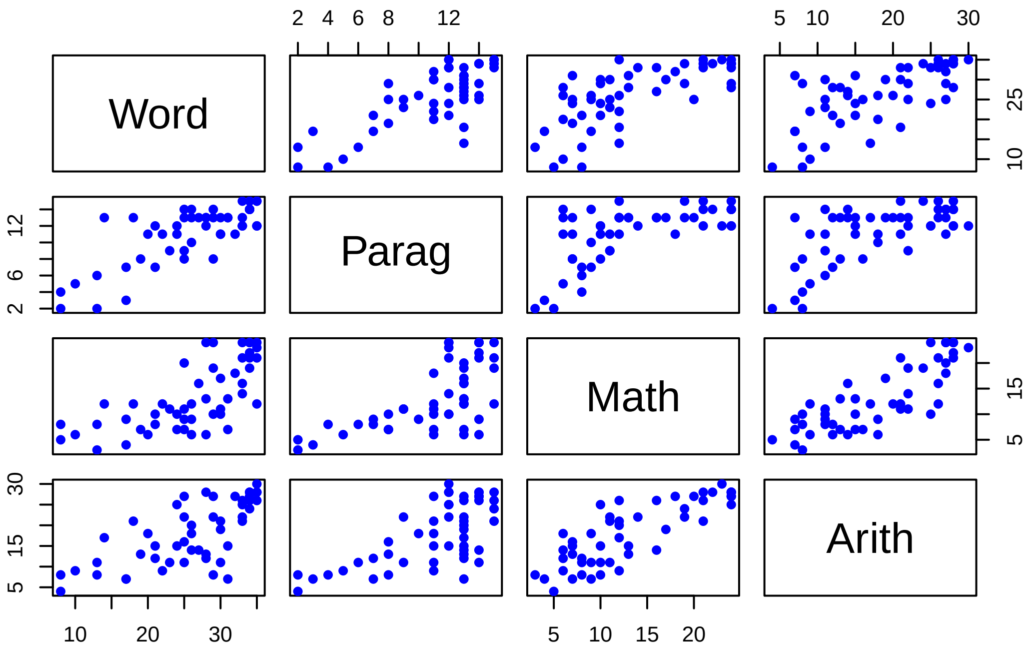
For simplicity we take a subset of 50 subjects.

```
set.seed(1)
data.AFQT <- data1[sample(nrow(data1), 50, replace=FALSE), c("Word","Parag", "Math", "Arith")]
#str(data.AFQT)
summary(data.AFQT)
```

```
##       Word           Parag            Math          Arith
##  Min.   : 8.00   Min.   : 2.00   Min.   : 3.0   Min.   : 4.00
##  1st Qu.:21.25   1st Qu.: 9.25   1st Qu.: 8.0   1st Qu.:11.25
##  Median :27.50   Median :12.00   Median :12.0   Median :18.00
##  Mean   :25.88   Mean   :11.04   Mean   :13.3   Mean   :18.02
##  3rd Qu.:32.75   3rd Qu.:13.00   3rd Qu.:19.0   3rd Qu.:25.75
##  Max.   :35.00   Max.   :15.00   Max.   :24.0   Max.   :30.00
```

We expect the four test scores correlated each other.

```
pairs(data.AFQT, pch=16, col="blue")    # shows pairwise relationship between two sets of scores
```




```
#rownames(data.AFQT)  # label for each person
rownames(data.AFQT) <-  paste("p", seq(1:nrow(data.AFQT)), sep="")  # reassign everyone's labels to be 
#rownames(data.AFQT)
```

2) Scaling and centering variables

```
sapply(data.AFQT, sd)  # sd's for each test
```

```
##     Word    Parag     Math    Arith
## 7.555617 3.457969 6.446800 7.528151
```

```r
sapply(data.AFQT, mean) # means for each test
```

```
##  Word Parag  Math Arith
## 25.88 11.04 13.30 18.02
```

```r
#colMeans((data.AFQT))
```

```r
# What are correlations?
#var(data.AFQT)  # covariances
cor(data.AFQT)
```

```
##            Word     Parag      Math     Arith
## Word  1.0000000 0.7766120 0.6949985 0.6964630
## Parag 0.7766120 1.0000000 0.6320325 0.6600638
## Math  0.6949985 0.6320325 1.0000000 0.8257461
## Arith 0.6964630 0.6600638 0.8257461 1.0000000
```

Each test score has its own variance. Sometimes the variables of interest may have different units. We can center the mean to 0 and set the sd to 1 for each test by subtracting the score means and dividing the sd for each test. Then the cor and var will be the same.

```r
data.AFQT.scale <- scale(data.AFQT, center=TRUE, scale=TRUE)  #default
#is.matrix(data.AFQT.scale) # it turns a data frame to a matrix
```

Now the mean of each test will be 0, and the var matrix will be same as the cor matrix.

```r
data.AFQT.scale <- as.data.frame(data.AFQT.scale)
colMeans(data.AFQT.scale)  # all zeros
```

```
##          Word         Parag          Math         Arith
##  1.437739e-16  2.353673e-16 -1.310063e-16  4.440892e-17
```

```r
sapply(data.AFQT.scale, sd)  # all 1's
```

```
##  Word Parag  Math Arith
##     1     1     1     1
```

```r
# Now the var matrix and cor matrix are the same after scaled.
var(data.AFQT.scale)
```

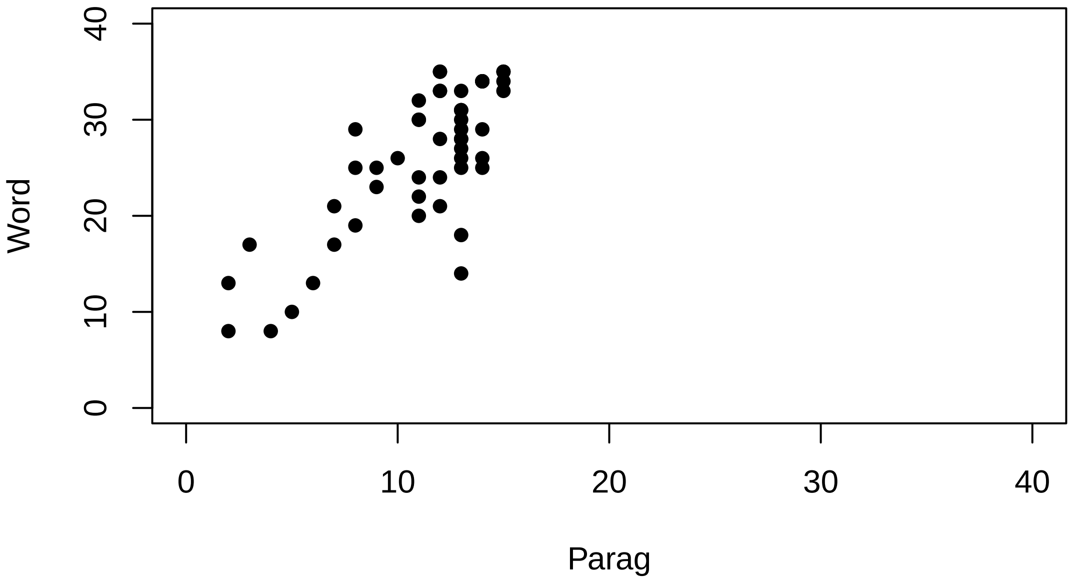
```
##            Word     Parag      Math     Arith
## Word  1.0000000 0.7766120 0.6949985 0.6964630
## Parag 0.7766120 1.0000000 0.6320325 0.6600638
## Math  0.6949985 0.6320325 1.0000000 0.8257461
## Arith 0.6964630 0.6600638 0.8257461 1.0000000
```

```r
#cor(data.AFQT.scale)
#pairs(data.AFQT.scale, pch=16, col="blue")
```

3) Principal Components: dimension reduction

i) For simplicity let's first look at one pair of the scores

```r
# Parag and Word
par(mfrow=c(1,1))
plot(data.AFQT$Parag, data.AFQT$Word, pch=16,
     xlab="Parag",
     ylab="Word",
     xlim=c(0, 40),
     ylim=c(0, 40))
```



Questions:

i) How can we use one score which combines both Parag and Word linearly, such that it will give us the largest variance?

ii) Can we find a line which is closest to all the points?

This is equivalent to find

$$\phi_{11}$$

and

$$\phi_{21}$$

such that

$$Var(Z_1) = Var(\phi_{11} * X_1 + \phi_{21} * X_2)$$

is maximized with constraint

$$\phi_{11}^2 + \phi_{21}^2 = 1$$

Here X1 and X2 are centered and scaled Parag and Word scores.

$$X_1 = (Parg - mean(Parag))/sd(Parag)$$

$$X_2 = (Word - mean(Word))/sd(Word)$$

```
attach(data.AFQT)
mean(Parag)
```

```
## [1] 11.04
```

```
sd(Parag)
```

```
## [1] 3.457969
```

```
mean(Word)
```

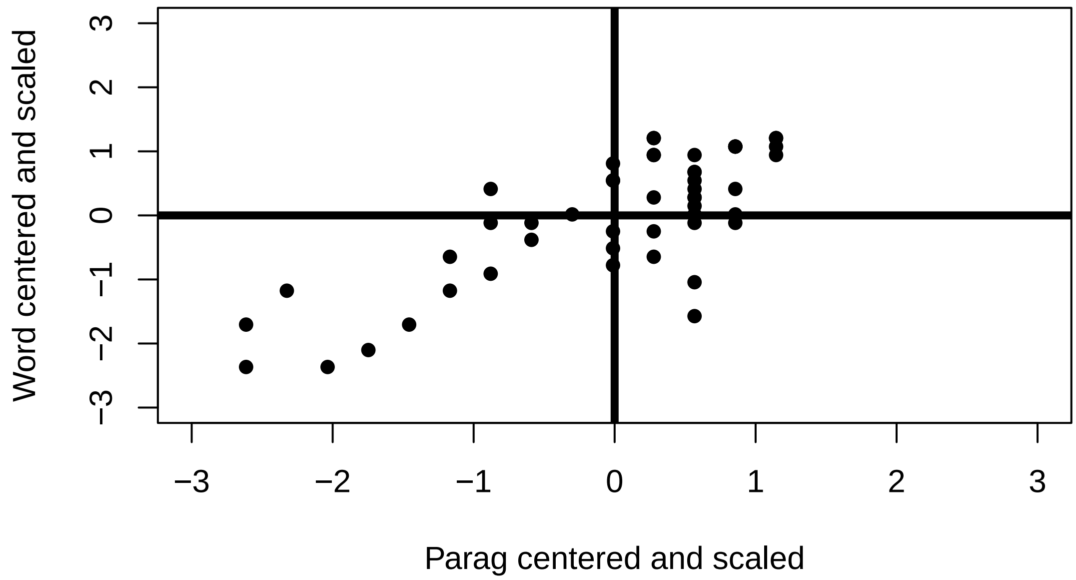
```
## [1] 25.88
```

```
sd(Word)
```

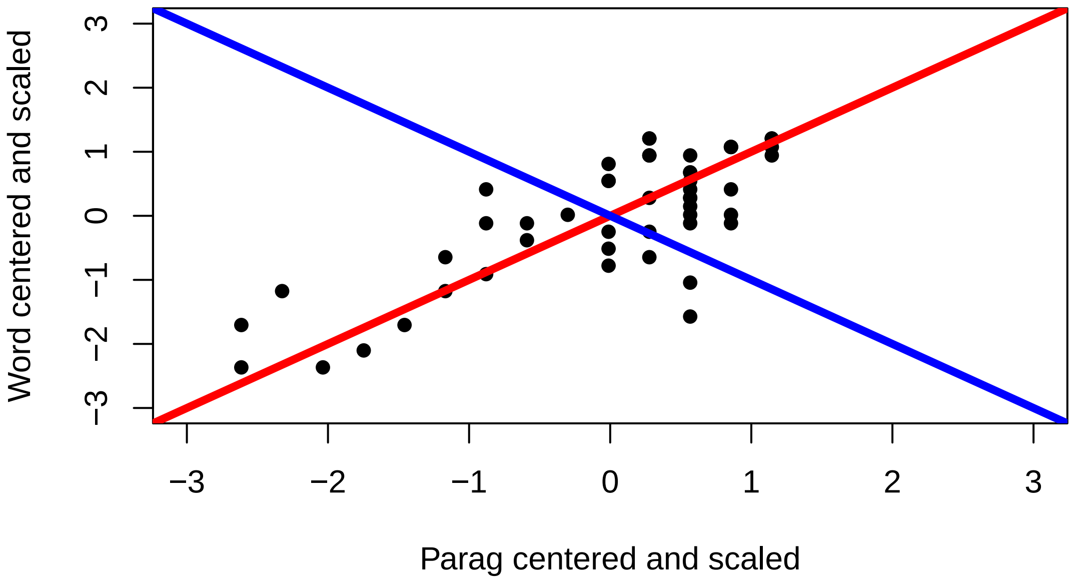
```
## [1] 7.555617
```

```
X1=(Parag - mean(Parag))/sd(Parag)
X2=(Word - mean(Word))/sd(Word)

# Same as
# scale(data.AFQT[, c("Parag", "Word") ], center=TRUE, scale=TRUE)
# By default center=TRUE and scale=TRUE
```

```
plot(X1, X2, pch=16, xlim=c(-3, 3), ylim=c(-3, 3),
     xlab="Parag centered and scaled",
     ylab="Word centered and scaled" )
abline(h=0, v=0, lwd=4)
```




```
par(mfrow=c(1,1))

plot(X1, X2, pch=16, xlim=c(-3, 3), ylim=c(-3, 3),
     xlab="Parag centered and scaled",
     ylab="Word centered and scaled" )
abline(0, 1, lwd=4, col="red")
abline(0, -1, lwd=4, col="blue")
```



ii) Terminology

Z_1: First principle component.

(ϕ_{11}, ϕ_{21}) is called the loadings.

The entire Z_1, one for each person, is called PC scores.

```
# Here we go: PCA
pc.parag.word <- prcomp(data.AFQT[, c("Parag", "Word")], scale=TRUE)
names(pc.parag.word)
```

```
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

```
pc.parag.word$rotation # Loadings = phi's
```

```
##              PC1        PC2
## Parag -0.7071068  0.7071068
## Word  -0.7071068 -0.7071068
```

```
phi_11 <- pc.parag.word$rotation[1,1]
phi_21 <- pc.parag.word$rotation[2,1]
```

```
Z1 <- phi_11 * X1 + phi_21 * X2  # PC scores. The two scores should be the same possibly by a different
max(abs(pc.parag.word$x[, 1]-Z1))   # Z1 and pc.parag.word$x[, 1] are the same.
```

```
## [1] 4.440892e-16
```

```
pc.parag.word$sdev  # sd(Z1) and sd(Z2)
```

```
## [1] 1.3328961 0.4726394
```

```
pc.parag.word$center  # means of the original scores
```

```
## Parag  Word
## 11.04 25.88
```

iii) Second Principal Component

Similar to the first Principal Component, we are now looking for ϕ_{12}, ϕ_{22}, such that

$$Z_2 = \phi_{12} * X1 + \phi_{22} * X2$$

where

$$Var(Z_2) = Var(\phi_{12} * X1 + \phi_{22} * X2)$$

is maximized subject to constraints $\phi_{12}^2 + \phi_{22}^2 = 1$, and Z_2 and Z_1 are orthogonal.

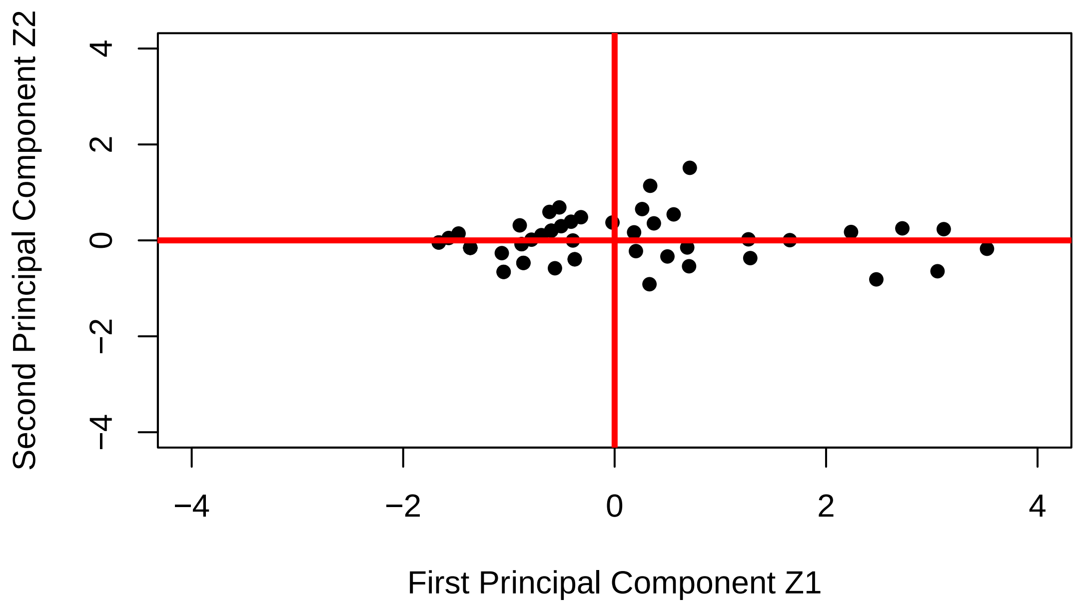
Z_2: Second principle component (PC2)

```
# The loadings
pc.parag.word$rotation # Loadings
```

```
##              PC1        PC2
## Parag -0.7071068  0.7071068
## Word  -0.7071068 -0.7071068
```

```
Z2 <- pc.parag.word$x[,2]
```

```
# Let us verify
plot(Z1, Z2, pch=16,
     xlab="First Principal Component Z1",
     ylab="Second Principal Component Z2",
     xlim=c(-4, 4),
     ylim=c(-4, 4))
abline(v=0, h=0, lwd=3, col="red")
```




```
# The sd's
pc.parag.word$sdev   # sd's of pc scores
```

```
## [1] 1.3328961 0.4726394
```

1) The principal plot is a rotation for the original X1, X2 plot
2) $$var(Z_1) = 1.777 > var(Z_2) = 0.223$$
3) Z_1 and Z_2 are orthogonal

```
cor(Z1, Z2) # is 0
```

```
## [1] -1.289369e-15
```

Principal Component of the four tests: Word, Math, Parag and Arith

Question:

1) How can we best capture the performance based on the four tests?
2) Can we come up with some sensible scores combining the four tests together?

PCs: Looking for a linear transformation of X1=Word, X2=Parag, X3=Math, and X4=Arith to have the max variance.

First Principal Component is

$$Z_1 = \phi_{11} * X_1 + \phi_{21} * X_2 + \phi_{31} * X_3 + \phi_{41} * X_4$$

such that $Var(Z_1)$ is maximized with $\sum \phi_{i1}^2 = 1$.

Second Principal Component is

$$Z_2 = \phi_{12} * X_1 + \phi_{22} * X_2 + \phi_{32} * X_3 + \phi_{42} * X_4$$

such that $Var(Z_2)$ is maximized, $\sum \phi_{i2}^2 = 1$, and Z_2 and Z_1 are uncorrelated (Orthogonal). This is the same as the $\{\phi_{i,1}\}$ and $\{\phi_{i,2}\}$s are orthogonal.

We keep going to obtain Z3, and Z4

1) Scale and Center each score

   To find sensible PC's, we recommend to

   - center each variable by subtracting its mean.
   - scale each variable by dividing its sd.
   - above two things can be achieved simultaneously by scale()
   - prcomp() has an option to scale or not

```
data.AFQT.scale <- scale(data.AFQT, center=TRUE, scale = TRUE)
summary(data.AFQT.scale)
```

```
##       Word              Parag              Math              Arith
##  Min.   :-2.3665   Min.   :-2.6143   Min.   :-1.5977   Min.   :-1.862343
##  1st Qu.:-0.6128   1st Qu.:-0.5176   1st Qu.:-0.8221   1st Qu.:-0.899291
##  Median : 0.2144   Median : 0.2776   Median :-0.2017   Median :-0.002657
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.000000
##  3rd Qu.: 0.9093   3rd Qu.: 0.5668   3rd Qu.: 0.8842   3rd Qu.: 1.026812
##  Max.   : 1.2070   Max.   : 1.1452   Max.   : 1.6597   Max.   : 1.591360
```

```
data.AFQT.scale <- as.data.frame(data.AFQT.scale)  # set it back as a data frame
names(data.AFQT.scale)
```

```
## [1] "Word"  "Parag" "Math"  "Arith"
```

```
attach(data.AFQT.scale)
```

```
## The following objects are masked from data.AFQT:
##
##     Arith, Math, Parag, Word
```

2) Use prcomp()

We input the original variables. BUT set scale=TRUE

```
pc.4 <- prcomp(data.AFQT, scale=TRUE)  # by default, center=True but scale=FALSE!!!
names(pc.4)
```

```
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

```r
summary(pc.4)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4
## Standard deviation     1.7730 0.6817 0.46996 0.41324
## Proportion of Variance 0.7859 0.1162 0.05522 0.04269
## Cumulative Proportion  0.7859 0.9021 0.95731 1.00000
```

```r
var(pc.4$x[, 1])
```

```
## [1] 3.143648
```

```r
# Loadings (directions)
round(pc.4$rotation, 5)
```

```
##            PC1      PC2      PC3      PC4
## Word  -0.50386  0.39736 -0.74849 -0.16729
## Parag -0.48672  0.60195  0.60712  0.17937
## Math  -0.50217 -0.51890 -0.09072  0.68581
## Arith -0.50701 -0.45881  0.25088 -0.68520
```

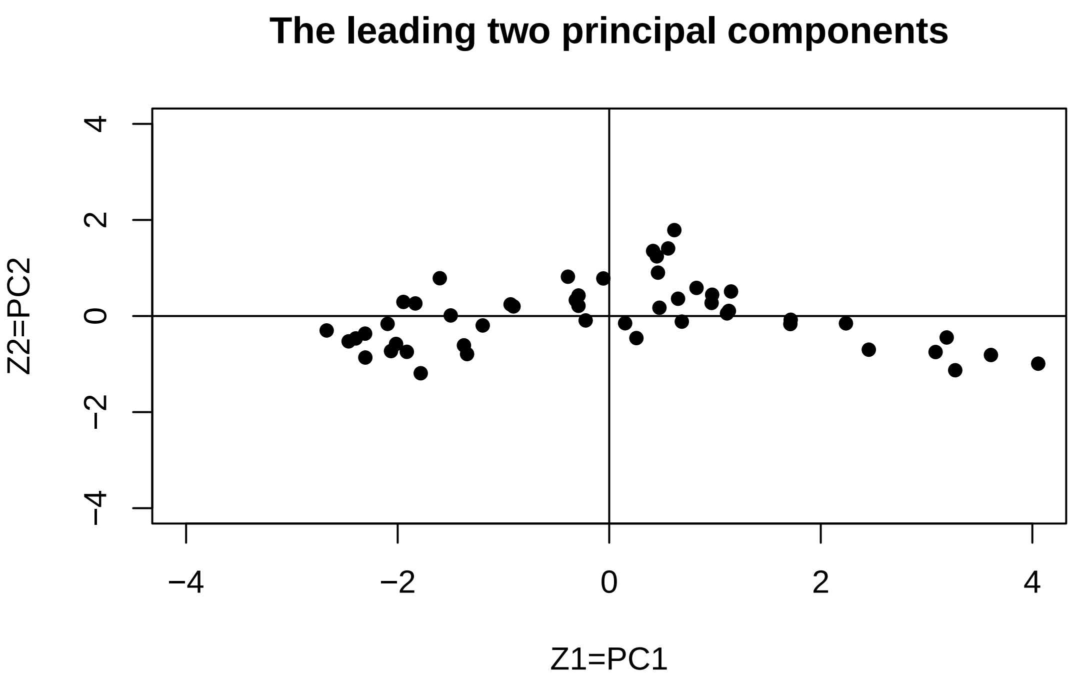
```r
# PC1 scores:
phi1 <- pc.4$rotation[,1]   # PC1 loadings, unique up to the sign
Z1.1 <- phi1[1]*Word+phi1[2]*Parag+phi1[3]*Math+phi1[4]*Arith
# Essentially it says that we should take the sum of the four test scores (after scaled)
# This is same as
Z1 <- pc.4$x[, 1]
max(abs(Z1.1-Z1))  # To convince you that two principal scores are the same.
```

```
## [1] 8.881784e-16
```

```r
# PC2 scores
Z2 <- pc.4$x[,2]
```

Interpretations of the pc scores:

```r
plot(pc.4$x[, 1], pc.4$x[,2 ], pch=16,
     xlim=c(-4, 4),
     ylim=c(-4, 4),
     main="The leading two principal components",
     xlab="Z1=PC1",
     ylab="Z2=PC2"
     )
abline(h=0, v=0)
```



Z1: total scores of the 4 tests

Z2: sum of the word and parg - sum of the math's

Remark:

i) Two scores Z1 and Z2 capture the main features of the four tests

ii) How much information we might have lost by using only the two PC scores?

NOT MUCH...

Properties of the pc scores:

i) var(Z1) > var(Z2) ....

```
c(sd(Z1), sd(Z2))    # they are reported in
```

```
## [1] 1.7730336 0.6817047
```

```
pc.4$sdev   # standard dev's of all PC's in a decreasing order
```

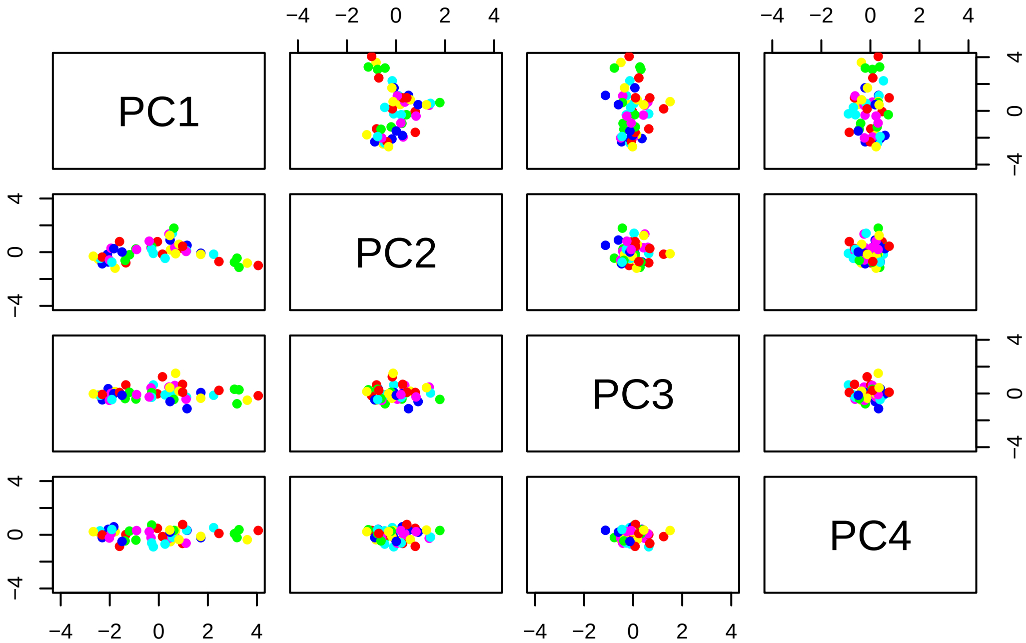
```
## [1] 1.7730336 0.6817047 0.4699631 0.4132375
```

ii) All 4 pc sores are uncorrelated.

```
cor(pc.4$x)    # cor's are 0
```

```
##                PC1           PC2          PC3          PC4
## PC1  1.000000e+00 -2.493174e-16 2.984269e-16 6.030205e-17
## PC2 -2.493174e-16  1.000000e+00 2.687433e-16 4.021501e-17
## PC3  2.984269e-16  2.687433e-16 1.000000e+00 2.537523e-16
## PC4  6.030205e-17  4.021501e-17 2.537523e-16 1.000000e+00
```

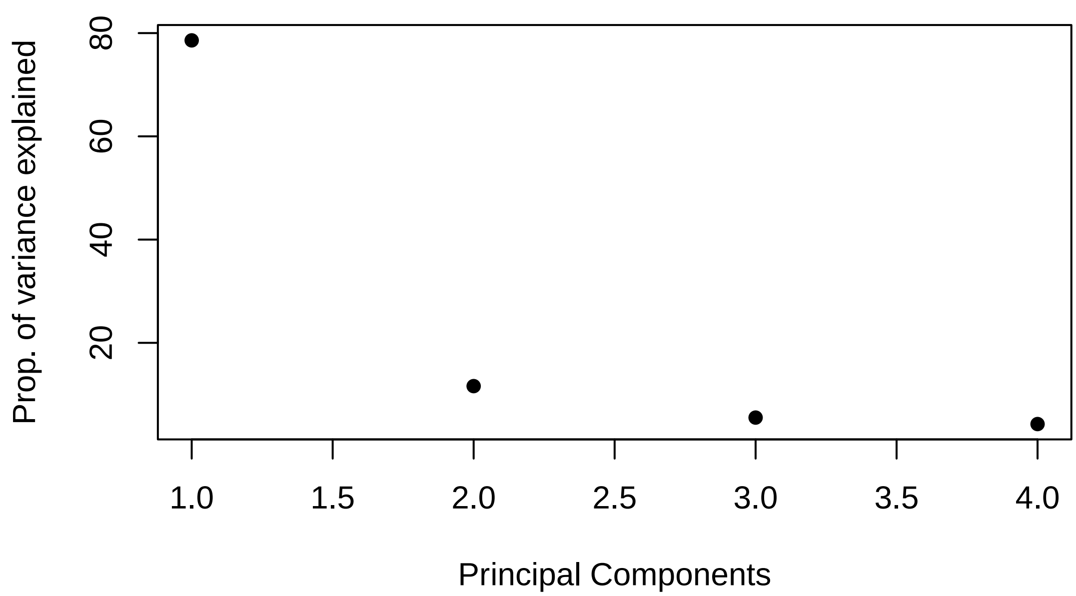
```
pairs(pc.4$x, xlim=c(-4, 4), ylim=c(-4, 4), col=rainbow(6), pch=16)
```



They are all pairwise uncorrelated!

iii) Proportion of variance explained (PVE)

```
pve.4 <- 100* (pc.4$sdev)^2/sum ((pc.4$sdev)^2)
plot(pve.4, pch=16,
     xlab="Principal Components",
     ylab="Prop. of variance explained")
```

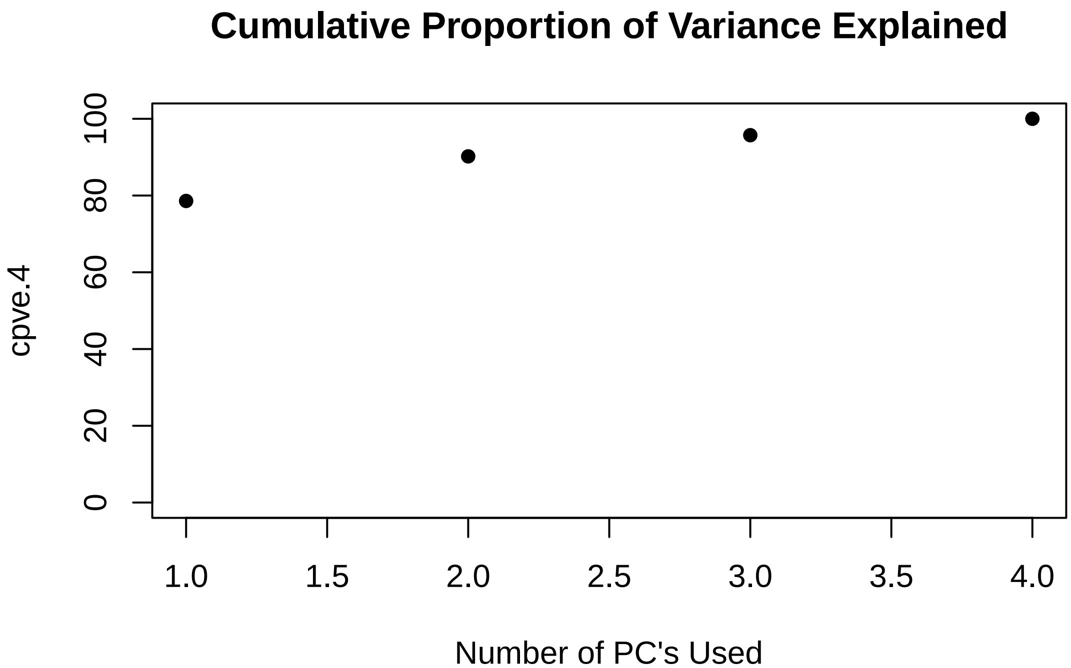


The leading component explains pve.4[1]=75% of the total variance.

Cumulative proportion of variance explained keeps track of the PVE including the first 1 PC, the first 2 PC's, and so on.

```
cpve.4 <- 100*cumsum((pc.4$sdev)^2)/4   # cumulative proportions of the variance explained by
  # the number of pc's used
```

```r
plot(seq(1:4), cpve.4, pch=16, ylim=c(0, 100),
     main="Cumulative Proportion of Variance Explained",
     xlab="Number of PC's Used")
```



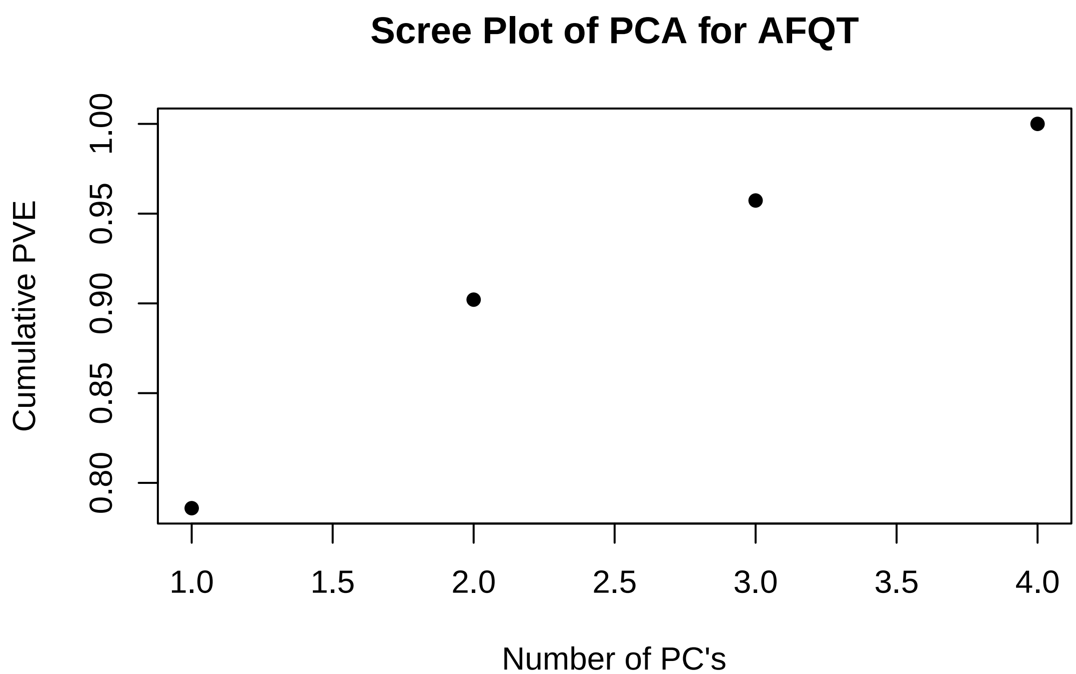

```r
names(summary(pc.4))
```

```
## [1] "sdev"       "rotation"   "center"     "scale"      "x"
## [6] "importance"
```

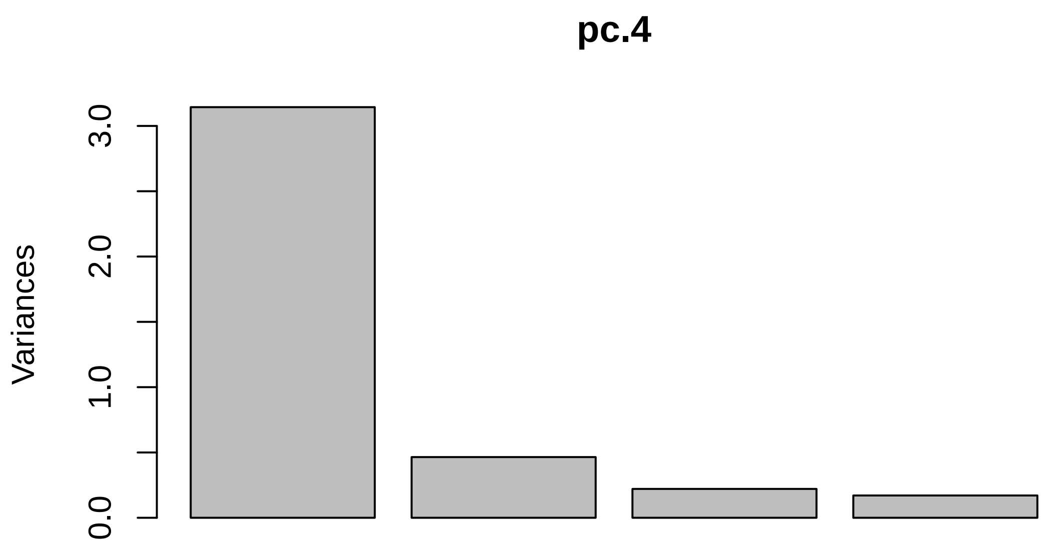
```r
summary(pc.4)$importance
```

```
##                             PC1       PC2       PC3       PC4
## Standard deviation     1.773034 0.6817047 0.4699631 0.4132375
## Proportion of Variance 0.785910 0.1161800 0.0552200 0.0426900
## Cumulative Proportion  0.785910 0.9020900 0.9573100 1.0000000
```

```r
# Scree plot of CPVE's
plot(summary(pc.4)$importance[3, ], pch=16,
     ylab="Cumulative PVE",
     xlab="Number of PC's",
     main="Scree Plot of PCA for AFQT")
```



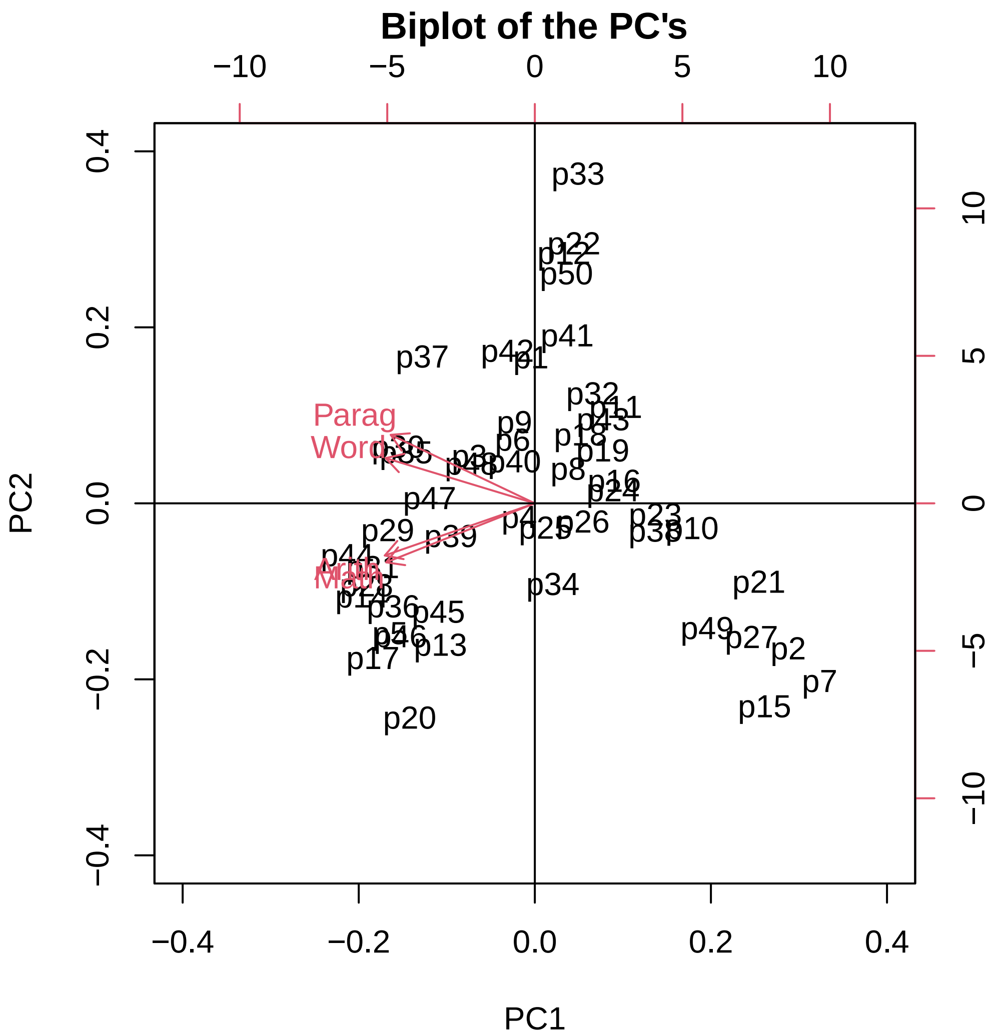

```
plot(pc.4) # variances of each pc
screeplot(pc.4) # var's each pc's
```



All the information is stored in summary(pc.4)

iv) biplot: visualize the PC scores together with the loadings of the original variables

```
lim <- c(-.4, .4)
biplot(pc.4, xlim=lim,
       ylim=lim,
       main="Biplot of the PC's")
abline(v=0, h=0)
```




```
# x-axis: PC1=Z1 (prop)
# y-axis: PC2=Z2
# top x-axis: prop to loadings for PC1
# right y-axis: prop to loadings for PC2
```

Summary: 1) To capture the main features of AFQT four scores we could use two summaries.

. Total scores (weighted)

. Difference of the math+arith and word+parag

Exercise: Perform PCA of all 10 tests

Does Gender play any roll in the test scores?

Part II: Calculation of the PC's

PCs are nothing but eigen vectors/values of COR(X1,X2,... Xp) (if scaled) or COV(X1,X2) (unscaled) PCA are eigenvectors of Cor matrix

```r
PC.eig <- eigen(cor(data.AFQT))
PC.eig
```

```
## eigen() decomposition
## $values
## [1] 3.1436481 0.4647213 0.2208653 0.1707652
##
## $vectors
##            [,1]       [,2]       [,3]       [,4]
## [1,] -0.5038640  0.3973614  0.7484907  0.1672921
## [2,] -0.4867169  0.6019472 -0.6071185 -0.1793694
## [3,] -0.5021654 -0.5189016  0.0907156 -0.6858146
## [4,] -0.5070086 -0.4588078 -0.2508773  0.6851995
```

```r
summary(PC.eig)
```

```
##         Length Class  Mode
## values   4     -none- numeric
## vectors 16     -none- numeric
```

```r
PC.eig$vectors   # Loadings
```

```
##            [,1]       [,2]       [,3]       [,4]
## [1,] -0.5038640  0.3973614  0.7484907  0.1672921
## [2,] -0.4867169  0.6019472 -0.6071185 -0.1793694
## [3,] -0.5021654 -0.5189016  0.0907156 -0.6858146
## [4,] -0.5070086 -0.4588078 -0.2508773  0.6851995
```

```r
PC.eig$values    # Variances of each PC's
```

```
## [1] 3.1436481 0.4647213 0.2208653 0.1707652
```

```r
# We use prcomp() here
PC <- prcomp(data.AFQT, scale=TRUE)
PC  # should be exactly same as PC's from eigen decomposition (up to the sign)
```

```
## Standard deviations (1, .., p=4):
## [1] 1.7730336 0.6817047 0.4699631 0.4132375
##
## Rotation (n x k) = (4 x 4):
##              PC1        PC2        PC3        PC4
## Word  -0.5038640  0.3973614 -0.7484907 -0.1672921
## Parag -0.4867169  0.6019472  0.6071185  0.1793694
## Math  -0.5021654 -0.5189016 -0.0907156  0.6858146
## Arith -0.5070086 -0.4588078  0.2508773 -0.6851995
```

```r
phi <- PC$rotation
phi
```

```
##              PC1        PC2        PC3        PC4
## Word  -0.5038640  0.3973614 -0.7484907 -0.1672921
## Parag -0.4867169  0.6019472  0.6071185  0.1793694
## Math  -0.5021654 -0.5189016 -0.0907156  0.6858146
## Arith -0.5070086 -0.4588078  0.2508773 -0.6851995
```

```r
cbind(PC.eig$vectors, PC$rotation) # Putting the first PC's together
```

```
##                                                          PC1        PC2
## Word  -0.5038640  0.3973614  0.7484907  0.1672921 -0.5038640  0.3973614
## Parag -0.4867169  0.6019472 -0.6071185 -0.1793694 -0.4867169  0.6019472
## Math  -0.5021654 -0.5189016  0.0907156 -0.6858146 -0.5021654 -0.5189016
## Arith -0.5070086 -0.4588078 -0.2508773  0.6851995 -0.5070086 -0.4588078
##              PC3        PC4
## Word  -0.7484907 -0.1672921
## Parag  0.6071185  0.1793694
## Math  -0.0907156  0.6858146
## Arith  0.2508773 -0.6851995
```

```r
# one from eigen-values???the other from prcomp(). They should be exactly the same (to the sign) and
# they are the same.
```

phi as we expected is an orthogonal unit matrix, i.e., the columns are uncorrelated with norm to be 1. Also,

$$inv(phi) = t(phi)!$$

This can be seen from

```r
phi %*% t(phi)   # matrix operator %*%. It should be an identity matrix
```

```
##                Word         Parag          Math         Arith
## Word   1.000000e+00 -1.494905e-16 -1.784226e-17 -8.159298e-17
## Parag -1.494905e-16  1.000000e+00  1.168337e-16  4.624411e-17
## Math  -1.784226e-17  1.168337e-16  1.000000e+00 -1.788495e-16
## Arith -8.159298e-17  4.624411e-17 -1.788495e-16  1.000000e+00
```

```r
round(phi %*% t(phi), 2) # better seen if we round it.
```

```
##       Word Parag Math Arith
## Word     1     0    0     0
## Parag    0     1    0     0
## Math     0     0    1     0
## Arith    0     0    0     1
```

Part III: Code for USArrests

PCA via prcomp

```
states=row.names(USArrests)
#states
names(USArrests)
```

```
## [1] "Murder"   "Assault"  "UrbanPop" "Rape"
```

```
apply(USArrests, 2, mean)
```

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

```
apply(USArrests, 2, var)
```

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

```
pr.out=prcomp(USArrests, scale=TRUE)
# By default, the prcomp() function centers the variables to have mean zero.
# By using the option scale=TRUE, we scale the variables to have standard deviation one.
```

```
names(pr.out)
```

```
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

```
pr.out$center
```

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

```
pr.out$scale
```

```
##    Murder   Assault  UrbanPop      Rape
##  4.355510 83.337661 14.474763  9.366385
```

```
pr.out$scale^2
```

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

```
# The center and scale components correspond to the means and standard
# deviations of the variables that were used for scaling prior to implementing PCA.
# The prcomp() function also outputs the standard deviation of each principal component
pr.out$sdev
```

```
## [1] 1.5748783 0.9948694 0.5971291 0.4164494
```

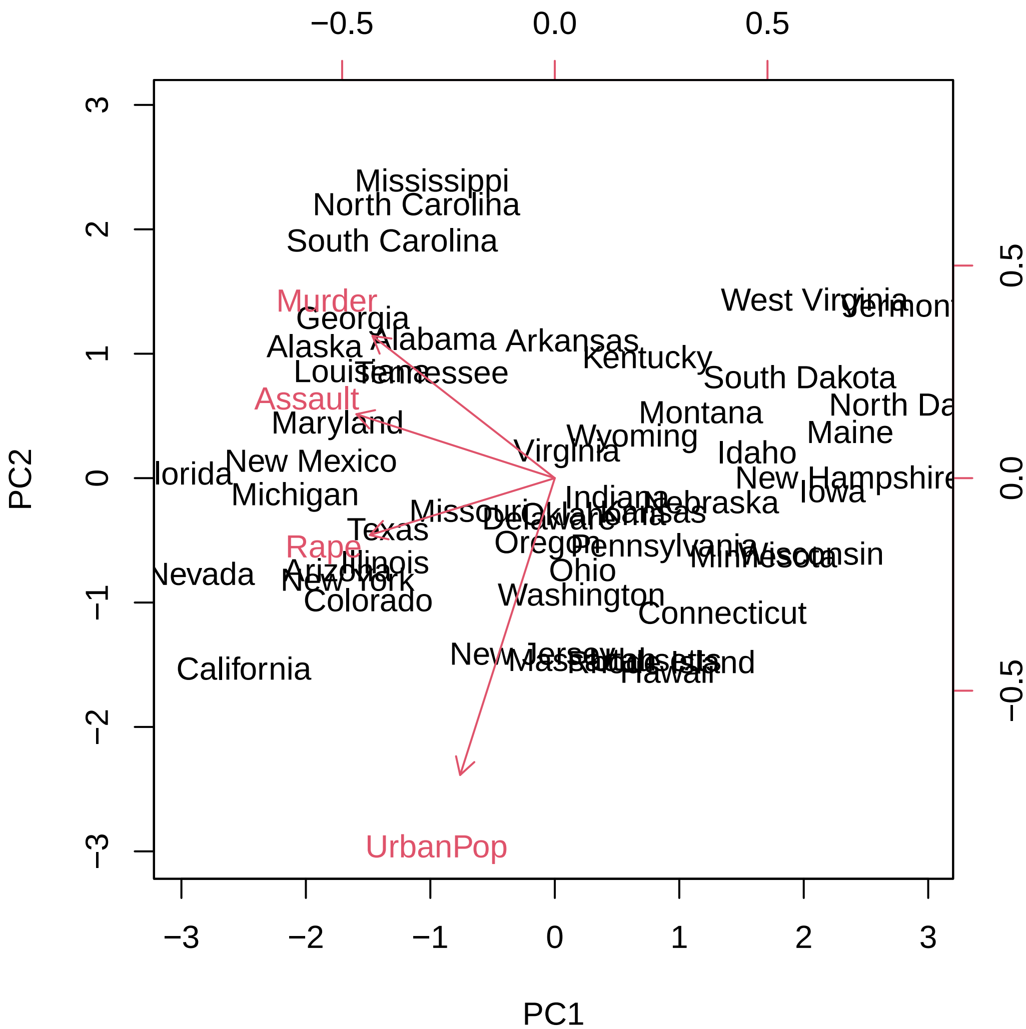
```r
# The rotation matrix provides the principal component loadings
pr.out$rotation
```

```
##                PC1        PC2        PC3         PC4
## Murder   -0.5358995  0.4181809 -0.3412327  0.64922780
## Assault  -0.5831836  0.1879856 -0.2681484 -0.74340748
## UrbanPop -0.2781909 -0.8728062 -0.3780158  0.13387773
## Rape     -0.5434321 -0.1673186  0.8177779  0.08902432
```

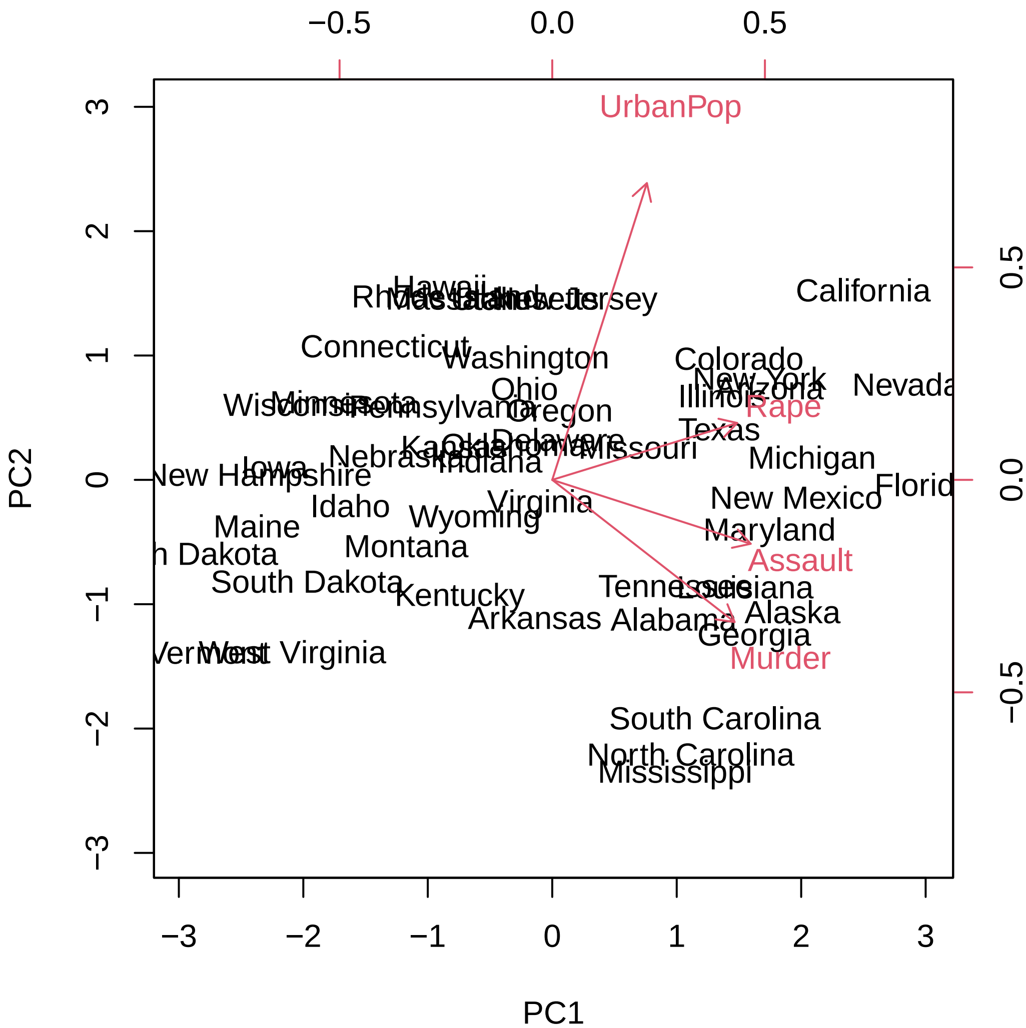
```r
# The kth column is the kth principal component score vector
dim(pr.out$x)
```

```
## [1] 50  4
```

```r
# The scale=0 argument to biplot() ensures that the arrows are scaled to
# represent the loadings
biplot(pr.out, scale=0)
```




```r
# Recall that the principal components are only unique up to a sign change
pr.out$rotation=-pr.out$rotation
pr.out$x=-pr.out$x
biplot(pr.out, scale=0)
```




```r
# The variance explained by each principal component is obtained by
pr.var=pr.out$sdev^2
pr.var
```

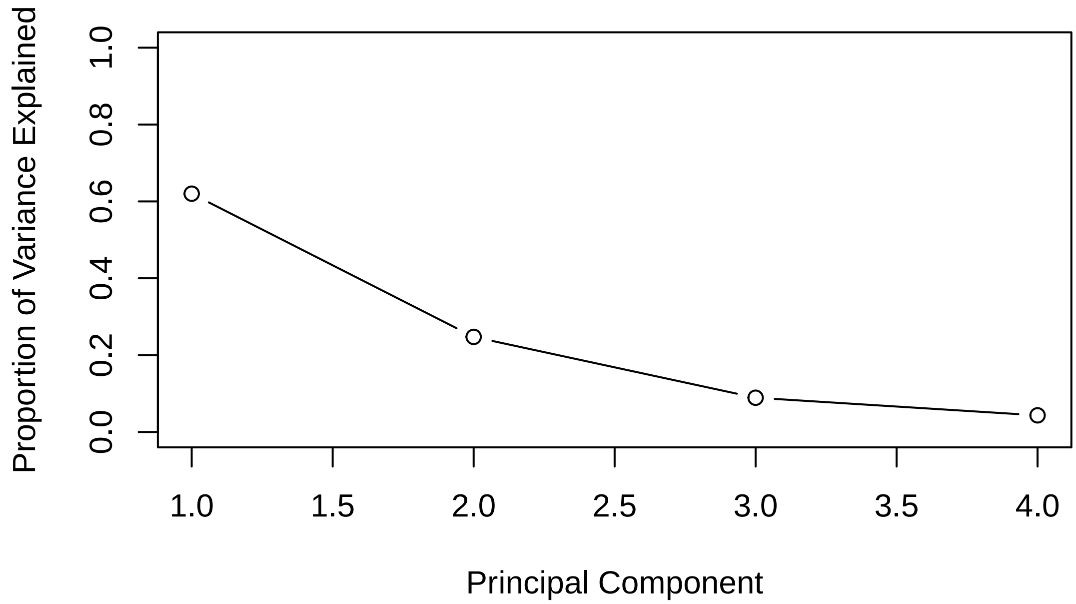
```
## [1] 2.4802416 0.9897652 0.3565632 0.1734301
```

```r
pve=pr.var/sum(pr.var)
pve
```

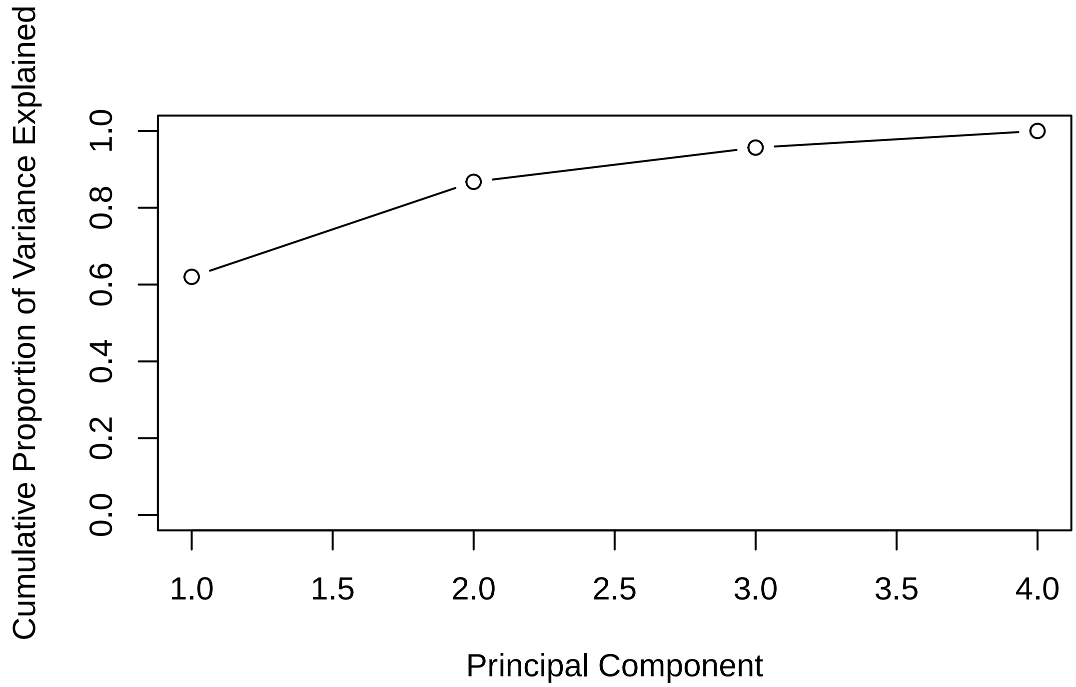
```
## [1] 0.62006039 0.24744129 0.08914080 0.04335752
```

To compute the proportion of variance explained by each principal component, we simply divide the variance explained by each principal component by the total variance explained by all four principal components

```r
plot(pve, xlab="Principal Component", ylab="Proportion of Variance Explained", ylim=c(0,1),type='b')
```




```
plot(cumsum(pve), xlab="Principal Component", ylab="Cumulative Proportion of Variance Explained", ylim=c
```



We can plot the PVE explained by each component, as well as the cumulative PVE

```
a=c(1,2,8,-3)
cumsum(a)
```

```
## [1]  1  3 11  8
```

PCA via svd

```
x.standardized <- scale(USArrests)
x.mean <- apply(x.standardized, 2, mean)
```

```r
x.sd <- apply(x.standardized, 2, sd)
x.svd <- svd(x.standardized)

x.score1 <- x.standardized %*% x.svd$v
x.score2 <- x.svd$u %*% diag(x.svd$d)
max(abs((-pr.out$x-x.score1)))
```

```
## [1] 0
```

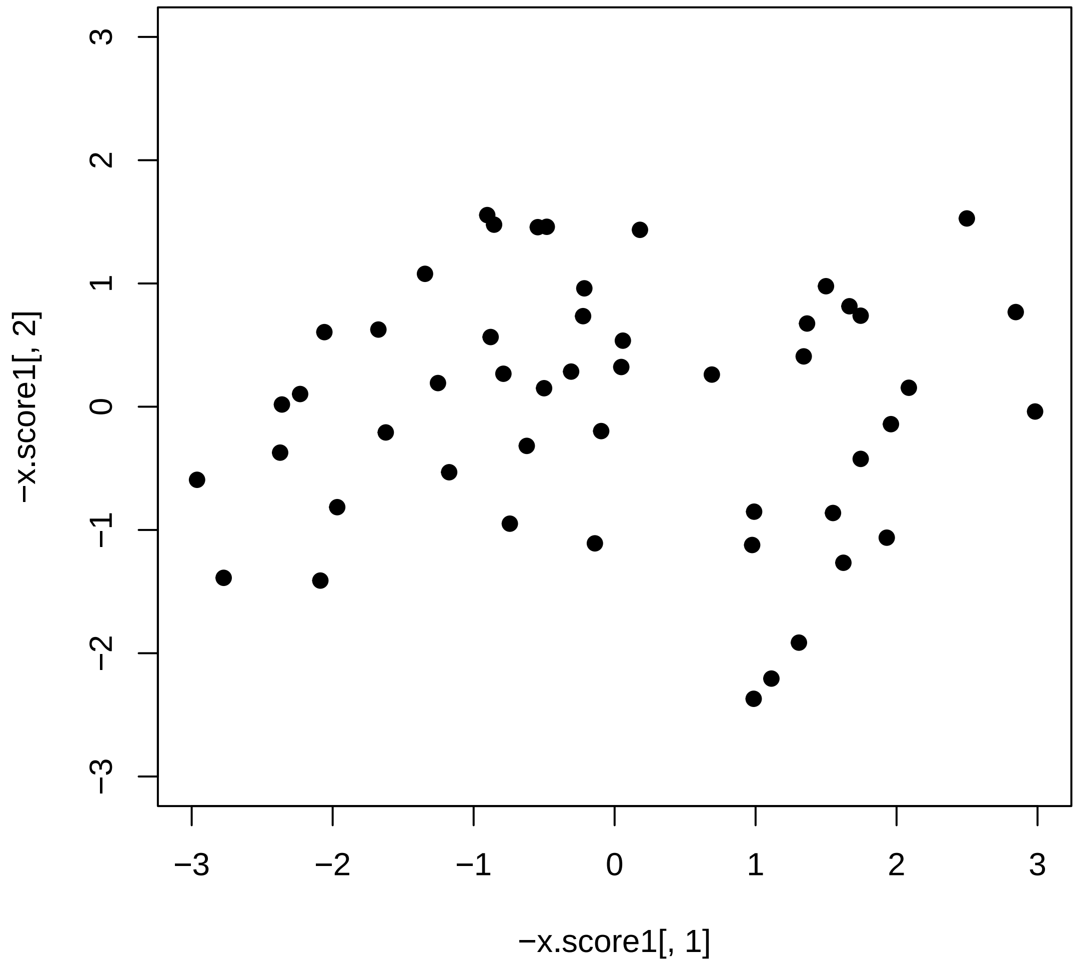
```r
max(abs((-pr.out$x-x.score2)))
```

```
## [1] 2.88658e-15
```

```r
x.svd$d / sqrt(nrow(x.score1)-1) - pr.out$sdev
```

```
## [1] 0 0 0 0
```

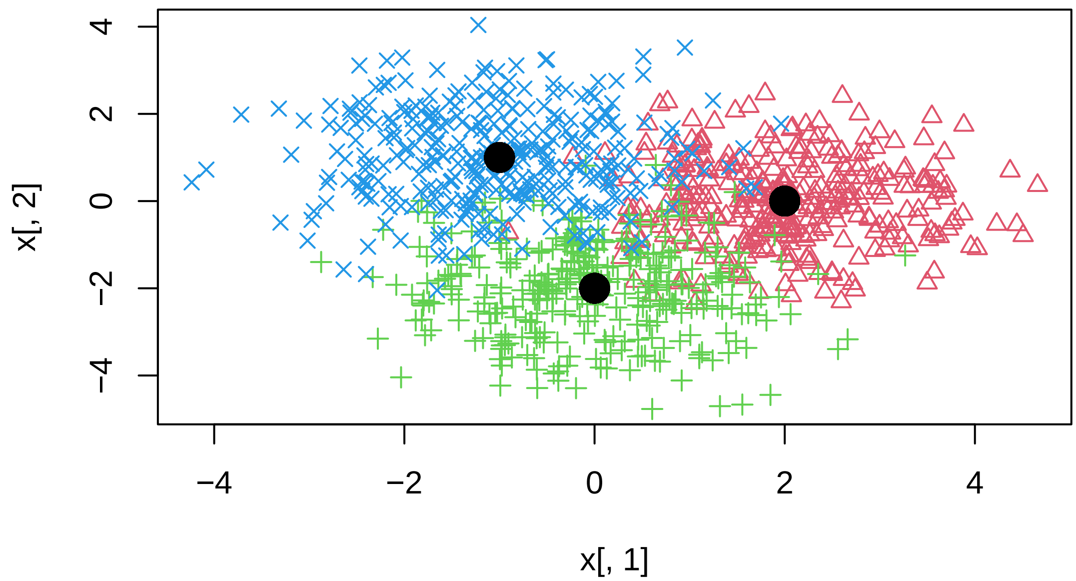
```r
plot(-x.score1[,1],-x.score1[,2],xlim=c(-3,3),ylim=c(-3,3), pch=19)
```



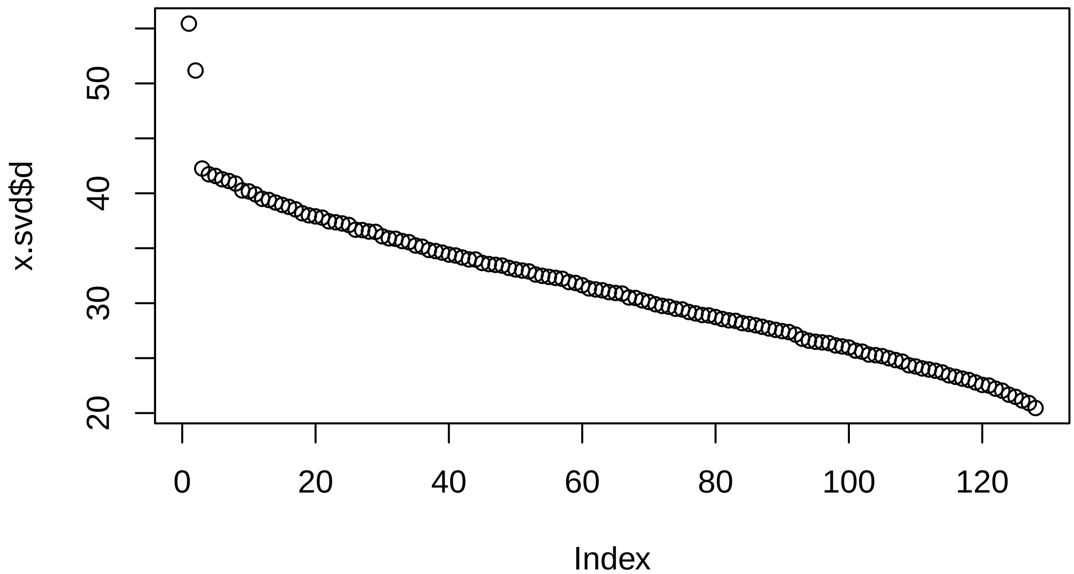

```r
#par(mfrow=c(1,2))
#biplot(pr.out, scale=0)
#plot(-x.score1[,1],-x.score1[,2],xlim=c(-3,3),ylim=c(-3,3), pch=19)
```

Part IV: Simulation Example

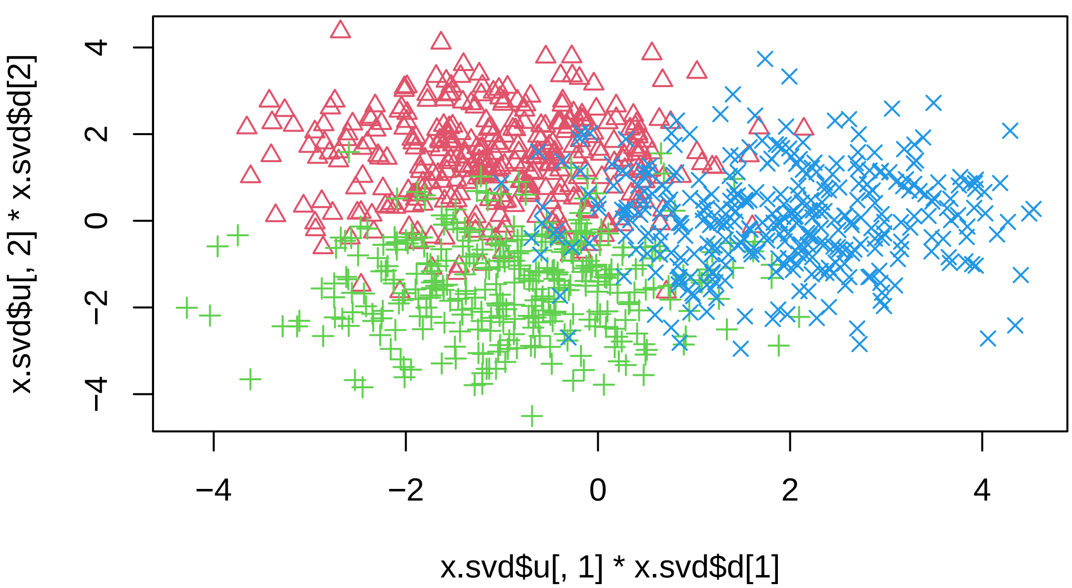
```
p <- 128
theta1 <- c(2,0,rep(0,p-2))
theta2 <- c(0,-2,rep(0,p-2))
theta3 <- c(-1,1,rep(0,p-2))
n <- 1002
# the mean matrix
mu <- rbind(matrix(rep(theta1,n/3),nrow=n/3,byrow=TRUE),
            matrix(rep(theta2,n/3),nrow=n/3,byrow=TRUE),
            matrix(rep(theta3,n/3),nrow=n/3,byrow=TRUE))
# observed data
x <- mu + matrix(rnorm(n*p),n)

# oracle
plot(x[,1],x[,2],col=rep(2:4,rep(n/3,3)),pch=rep(2:4,rep(n/3,3)))
points(mu[,1],mu[,2],pch=19,cex=2)
```



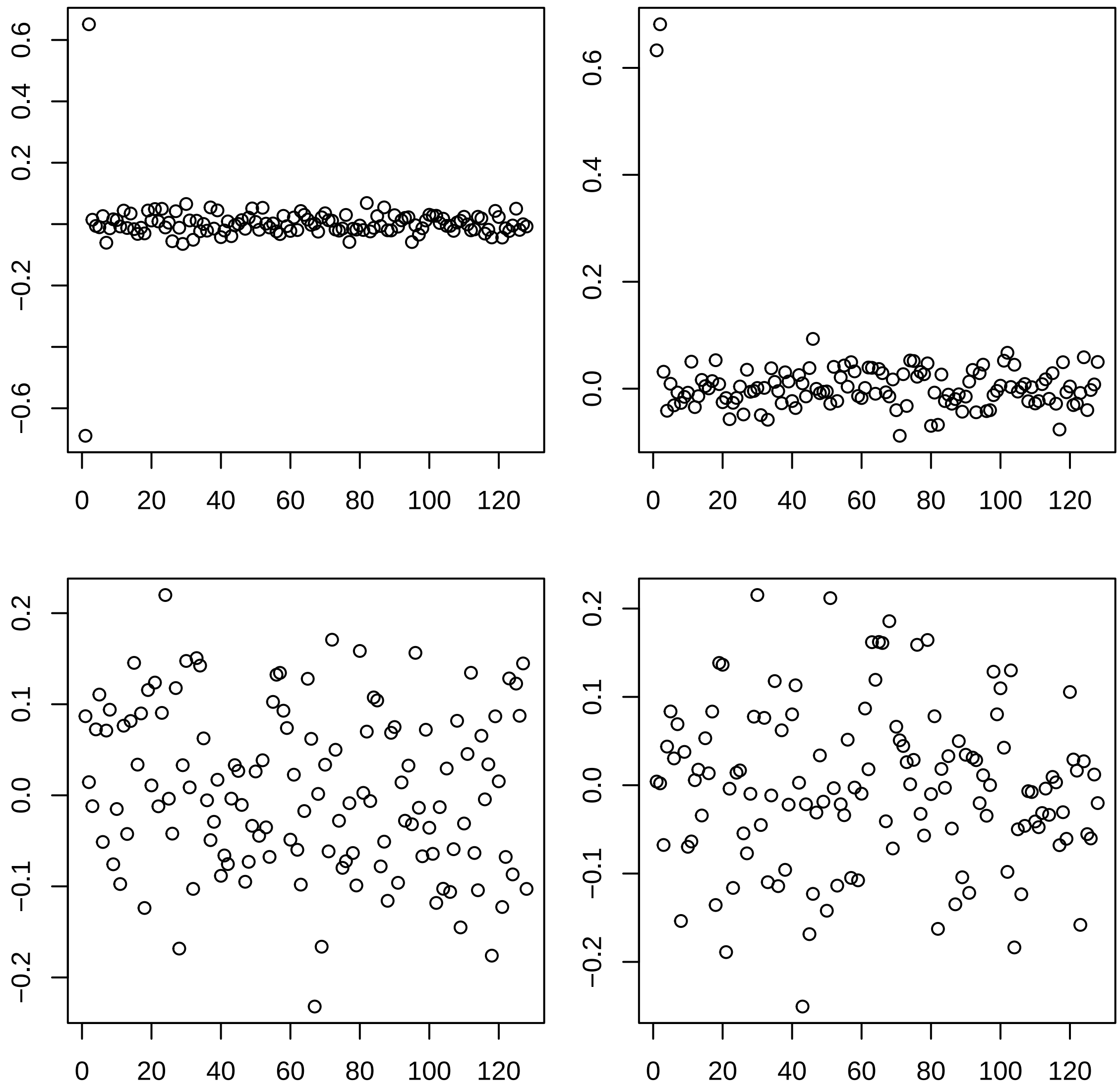

```
# svd, for singular value decomposition
mu.hat <- apply(x,2,mean)
x.centered <- x - rep(1,nrow(x))%*%t(mu.hat)
x.svd <- svd(x.centered)

# scree plot
plot(x.svd$d)
```



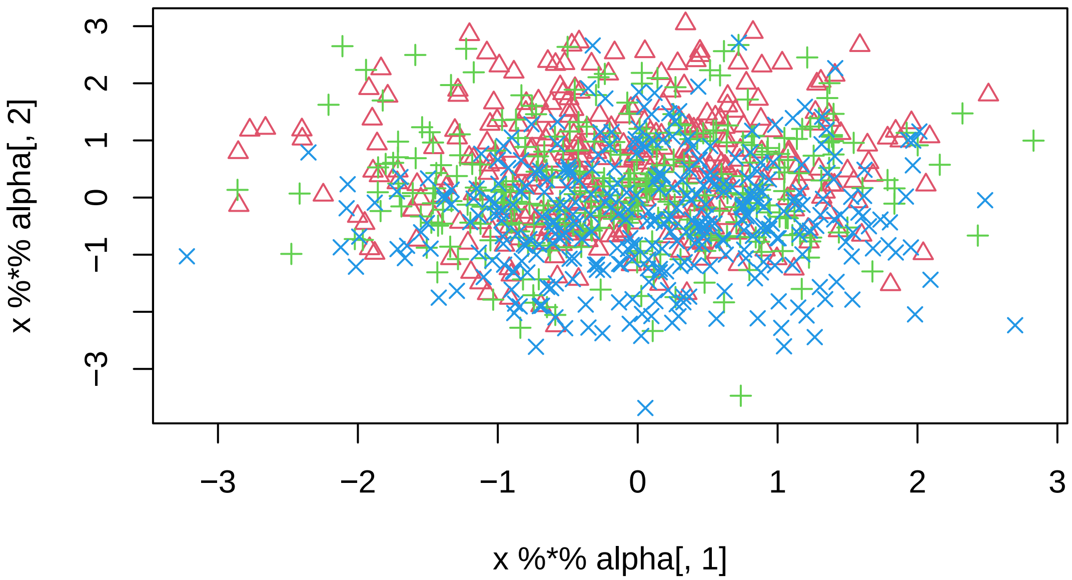

```
# plot PC projections
plot(x.svd$u[,1]*x.svd$d[1],x.svd$u[,2]*x.svd$d[2],col=rep(2:4,rep(n/3,3)),pch=rep(2:4,rep(n/3,3)))
```




```
# plot the loadings
par(mfrow=c(2,2), mar=c(2,2,2,1))
plot(x.svd$v[,1])
plot(x.svd$v[,2])
plot(x.svd$v[,3])
plot(x.svd$v[,4])
```




```
# generate random directions
alpha <- matrix(rnorm(p*2),ncol=2)
alpha <- qr.Q(qr(alpha))

# plot random projection
plot(x%*%alpha[,1],x%*%alpha[,2],col=rep(2:4,rep(n/3,3)),pch=rep(2:4,rep(n/3,3)))
```




```r
# 3 projections together
par(mfrow=c(1,3))
plot(x[,1],x[,2],
     col=rep(2:4,rep(n/3,3)),pch=rep(2:4,rep(n/3,3)),main="oracle projection")
points(mu[,1],mu[,2],pch=19,cex=2)
plot(x.svd$u[,1]*x.svd$d[1],x.svd$u[,2]*x.svd$d[2],
     col=rep(2:4,rep(n/3,3)),pch=rep(2:4,rep(n/3,3)),main="PCA projection")
plot(x%*%alpha[,1],x%*%alpha[,2],
     col=rep(2:4,rep(n/3,3)),pch=rep(2:4,rep(n/3,3)),main="random projection")
```