Solutions for ST340 Lab 2

2019–20

1: A simple singular value decomposition

(a) Generate a realization of a 4×5 Gaussian random matrix G.

```
G <- matrix(rnorm(20),4,5)
```

(b) Look at `?svd`.
(c) Set U, d, and V by using `svd`.

```
tmp <- svd(G)
U <- tmp$u; d <- tmp$d; V <- tmp$v
```

(d) Check that `G` is equal to `U%*%Sigma%*%t(V)` (to machine precision).

```
print(U%*%diag(d)%*%t(V))
```

```
##            [,1]       [,2]       [,3]       [,4]        [,5]
## [1,] -0.7579451  0.5540026  0.9456878 -0.5744063 -0.48113260
## [2,] -1.1766770 -0.1853125 -1.5629566 -1.1772905  0.41486328
## [3,] -0.8435489 -1.2455019 -0.9542814  1.0408114 -0.22422178
## [4,]  0.2874870 -1.4197076 -0.4955706 -0.7940780 -0.05562674
```

```
print(G)
```

```
##            [,1]       [,2]       [,3]       [,4]        [,5]
## [1,] -0.7579451  0.5540026  0.9456878 -0.5744063 -0.48113260
## [2,] -1.1766770 -0.1853125 -1.5629566 -1.1772905  0.41486328
## [3,] -0.8435489 -1.2455019 -0.9542814  1.0408114 -0.22422178
## [4,]  0.2874870 -1.4197076 -0.4955706 -0.7940780 -0.05562674
```

```
all.equal(G, U%*%diag(d)%*%t(V))
```

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

(e) Plot the singular values.

```
plot(d)
```

(f) Compute G_2, the 2-rank approximation of G, and also compute $||G - G_2||_F$.

```
G_2 <- U[,1:2]%*%diag(d[1:2])%*%t(V[,1:2])

print(norm(G-G_2,type='F'))
```

```
## [1] 1.858526
```

(g) Does the value agree with the theory?

```
print(sqrt(sum((G-G_2)^2)))
```

```
## [1] 1.858526
```

```
print(sqrt(sum(d[3:4]^2)))
```

```
## [1] 1.858526
```

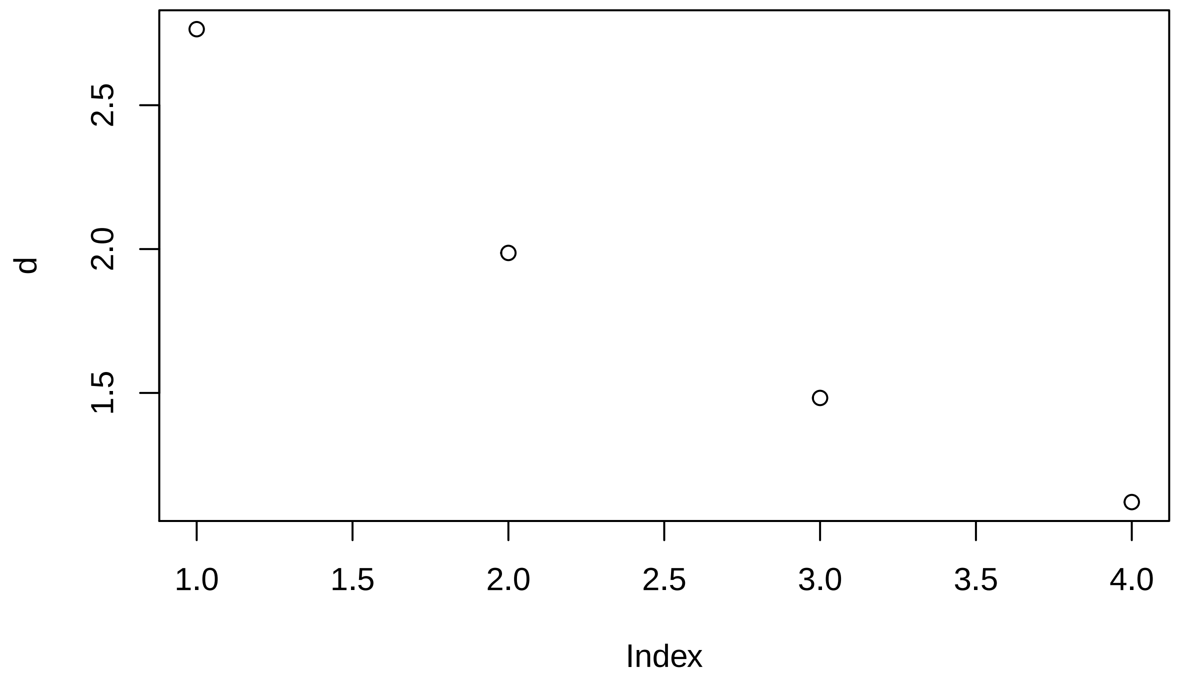


Figure 1: Q1(e) SVD of G

2: Image compression via the singular value decomposition

```r
load("pictures.rdata")
source("svd.image.compression.R")
```

Take a look at `svd.image.compression.R` and understand what the code is doing. Then run `image.compression()` here to see how well we can compress our images.

```r
image.compression()
```

Plot side-by-side the original image and the singular values

```r
res <- image.compress.param(1)
par(mfrow=c(1,2))
viewImage(res$mtx)
plot(res$svd$d)
abline(h=0,lty=2)
```

View the original and compressed image side-by-side

```r
k <- 10
compressedImage <- compute.compression(k, res$p, res$mtx, res$svd)
```

```
## [1] "approximation.error = 41.0683994323561"
## [1] "approximation.error.theory = 41.0683994323565"
```

```r
par(mfrow=c(1,2))
if (!is.null(compressedImage)) {
  viewImage(res$mtx)
  viewImage(compressedImage)
}
```

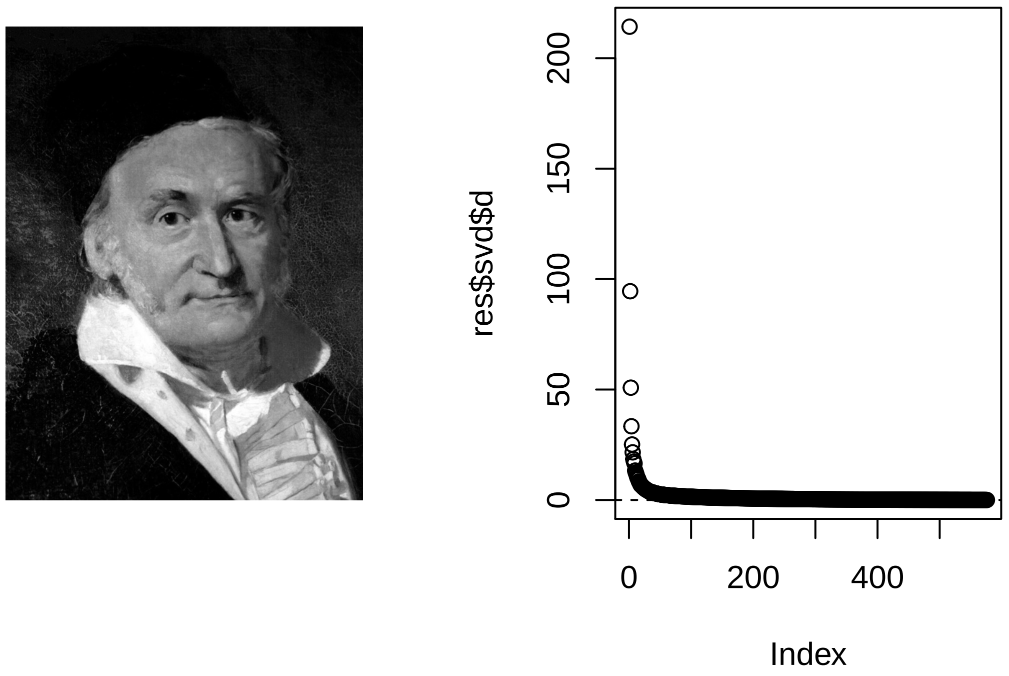


Figure 2: Singular values of the image

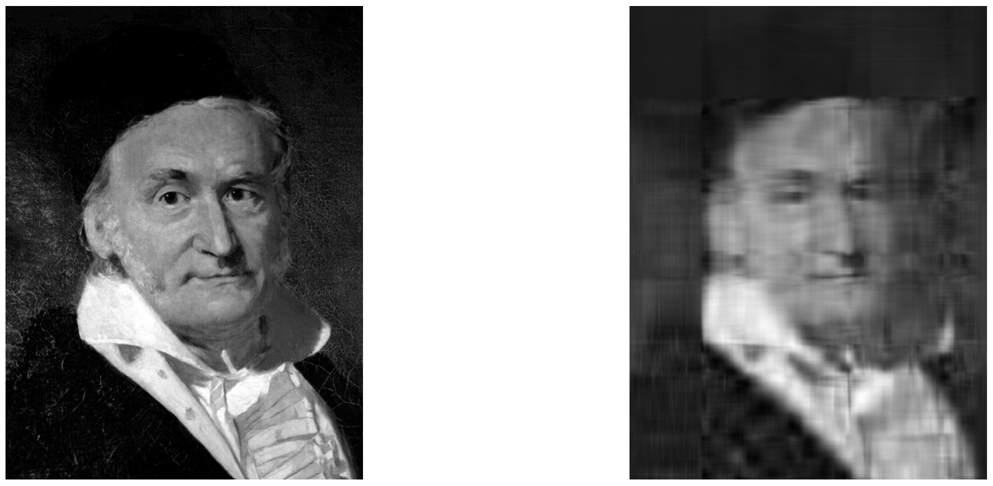


Figure 3: Comparison of (a) original; and (b) compressed images

3: PCA: Crabs

(a) Load the MASS library to access the crabs data.

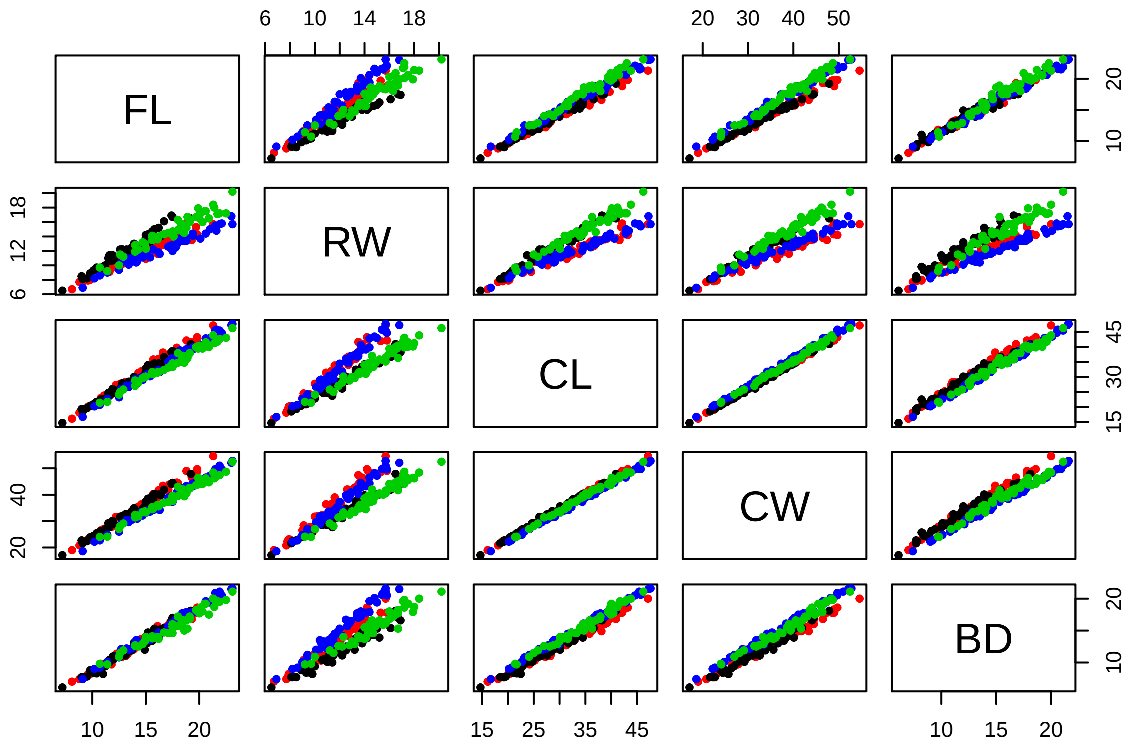
```
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 3.5.2
```

(b) Read `?crabs`.

(c) Read in the FL, RW, CL, CW, and BD measurements.

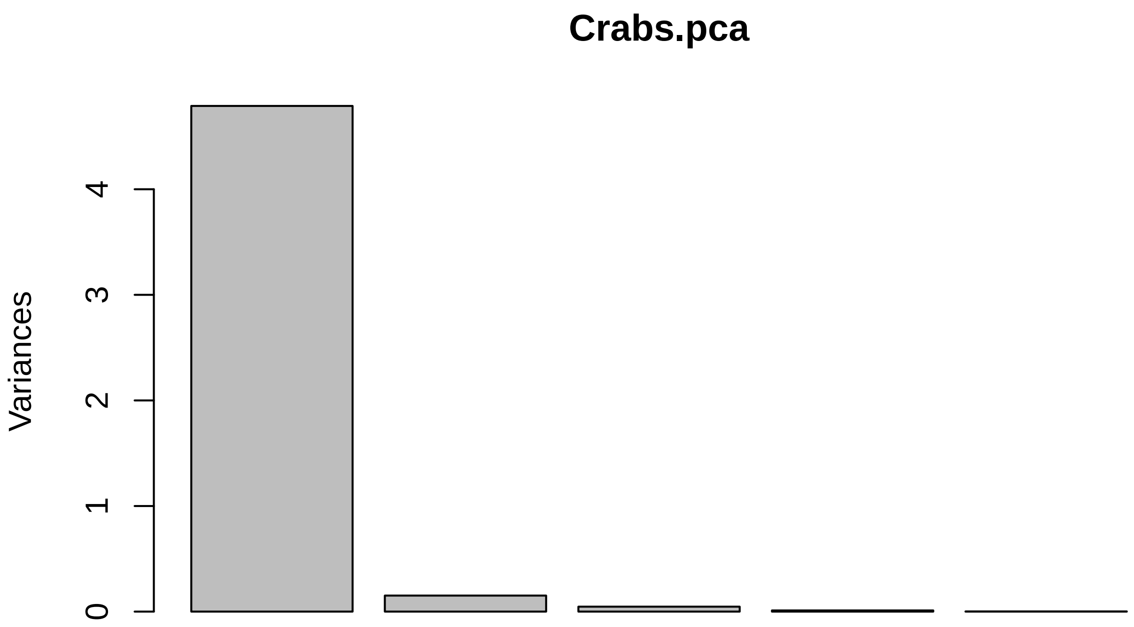
```
Crabs <- crabs[,4:8]
Crabs.class <- factor(paste(crabs[,1],crabs[,2],sep=""))
plot(Crabs,col=Crabs.class,pch=20)
```



(d) Read `?prcomp` and use it to obtain the principal components of a centred and scaled version of `Crabs`. Call the output of `prcomp` '`Crabs.pca`'.

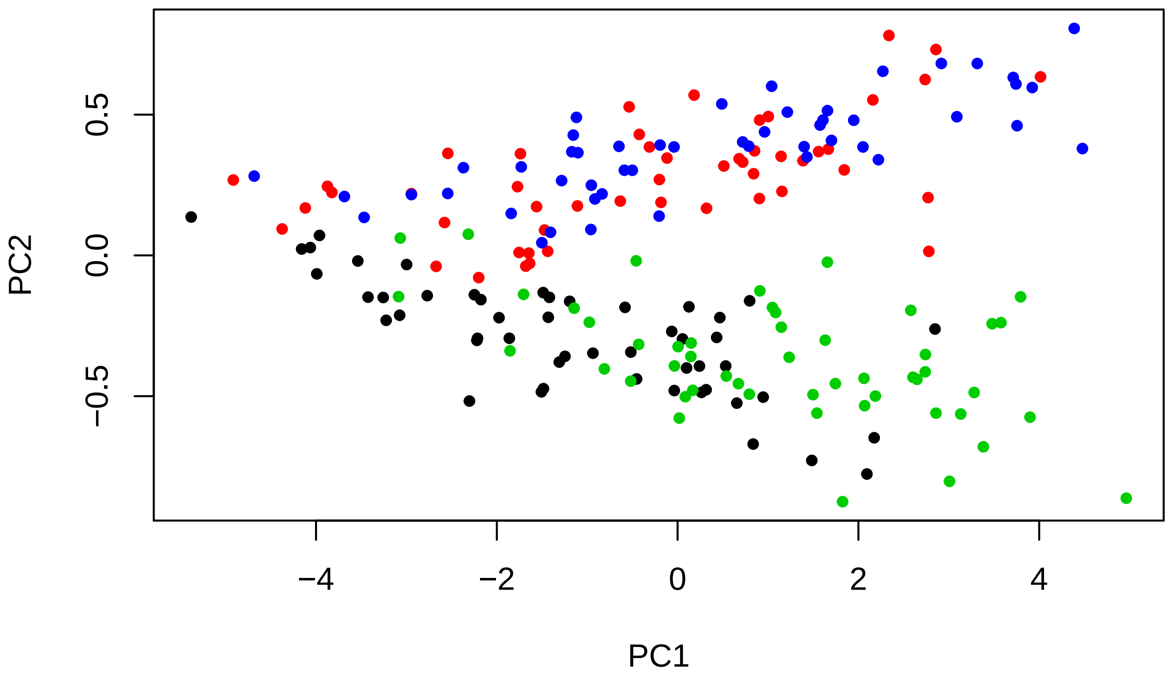
(e) If you `plot(Crabs.pca)` it visualizes the variances associated with the components.

```
Crabs.pca <- prcomp(Crabs,scale.=TRUE)
plot(Crabs.pca)
```



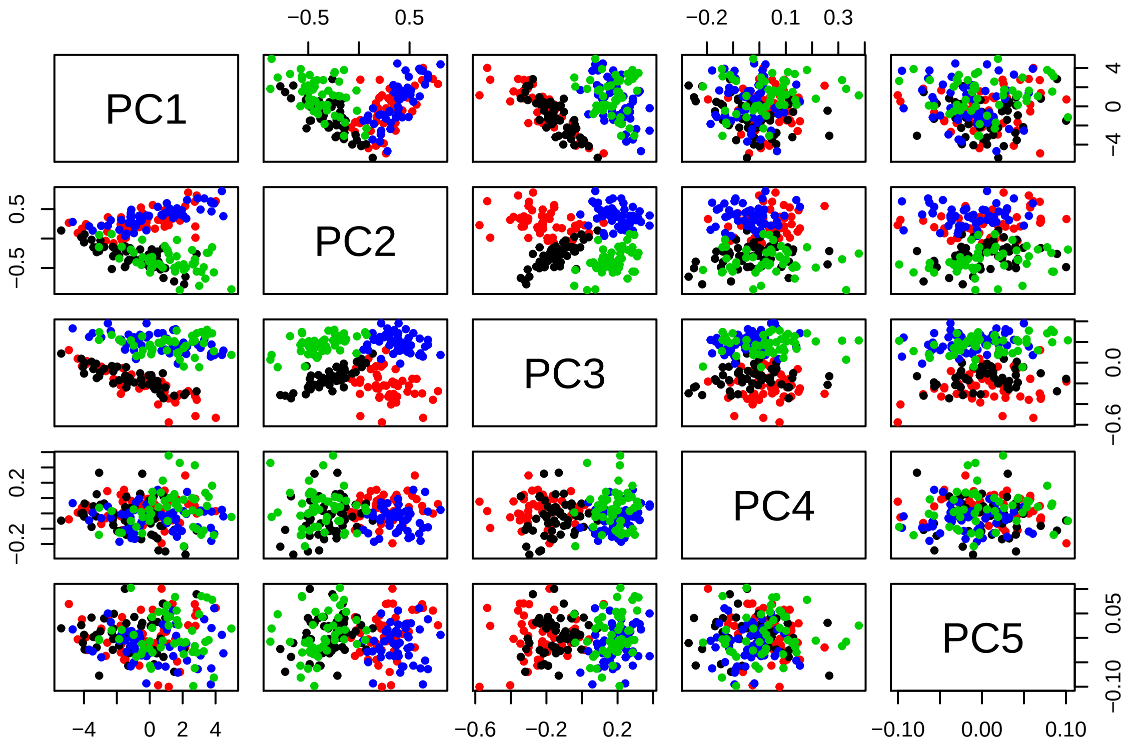
(f) Plot PC2 against PC1.

```
plot(Crabs.pca$x[,1:2],col=Crabs.class,pch=20)
```

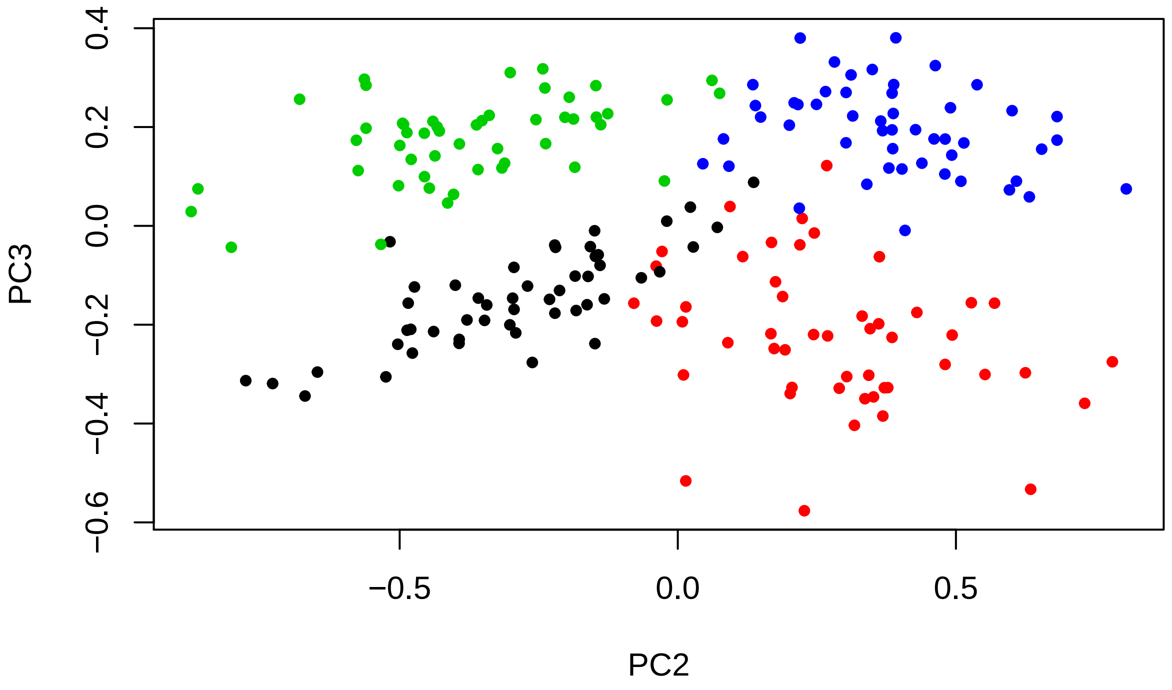


(g) Read `?pairs` and use it to find a pair of components with good separation of the classes.

```
pairs(Crabs.pca$x[,1:5],col=Crabs.class,pch=20)
```



```r
plot(Crabs.pca$x[,2:3],col=Crabs.class,pch=20)
```



(h) Read `?scale`. Check that you can obtain the principal components by using the singular value decomposition on a centred and scaled version of `Crabs`.

```r
scaledCrabs <- scale(Crabs)
Crabs.svd <- svd(scaledCrabs)
print(Crabs.svd$v - Crabs.pca$rotation)
```

```
##     PC1 PC2 PC3 PC4 PC5
## FL   0   0   0   0   0
## RW   0   0   0   0   0
## CL   0   0   0   0   0
## CW   0   0   0   0   0
## BD   0   0   0   0   0
```

```
Crabs.pcs <- scaledCrabs%*%Crabs.svd$v
print(norm(Crabs.pcs - Crabs.pca$x))
```

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

4: PCA: Viruses

This is a dataset on 61 viruses with rod-shaped particles affecting various crops (tobacco, tomato, cucumber and others) described by Fauquet *et al.* (1988) and analysed by Eslava-Gómez (1989). There are 18 measurements on each virus, the number of amino acid residues per molecule of coat protein.

```
load("viruses.rdata")
```

(a) Obtain the principal components of a centred and scaled version of allviruses.

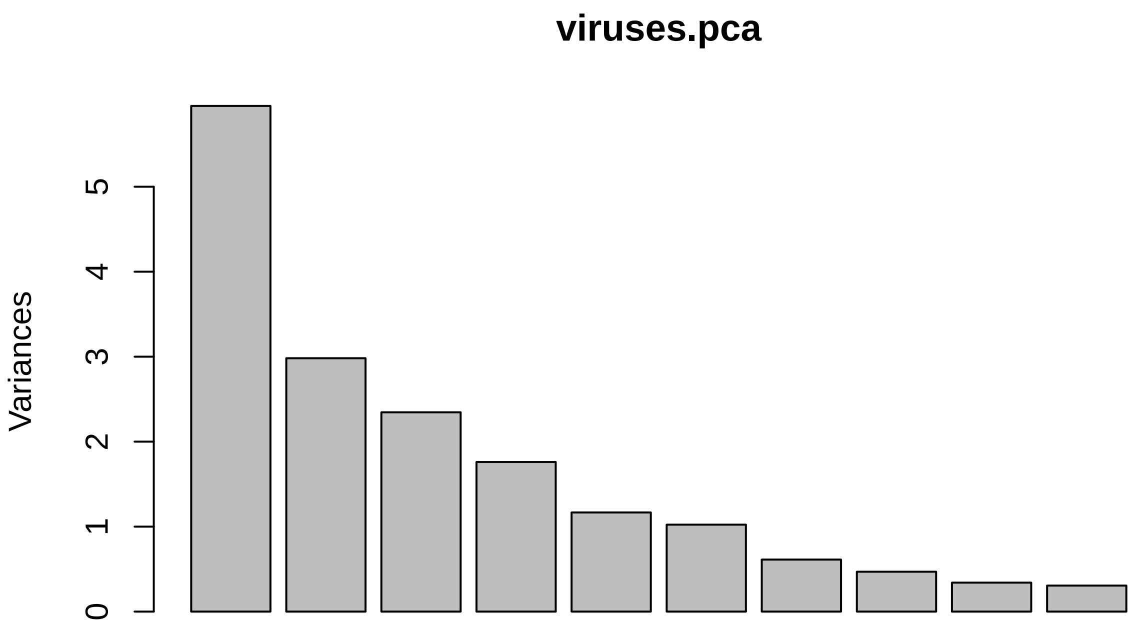
```
X <- scale(allviruses)
viruses.pca <- prcomp(X)

groups <- rep(0,61)
groups[1:3] <- 1
groups[4:9] <- 2
groups[10:48] <- 3
groups[49:61] <- 4
group.names <- c("Hordeviruses","Tobraviruses","Tobamoviruses","furoviruses")
```

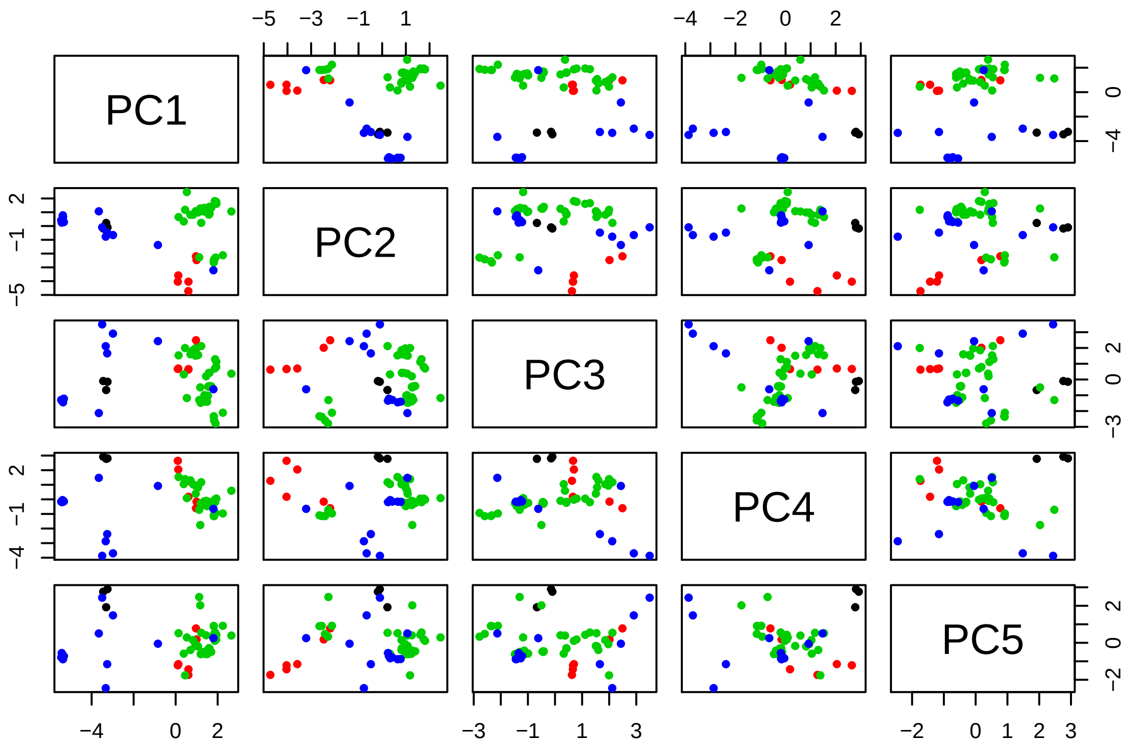
If you colour by groups (i.e. `col=groups` in plot) then black is horde, red is tobra, green is tobamo, blue is furo.

(b) Do the principal components show some separation between the viruses?

```
plot(viruses.pca)
```

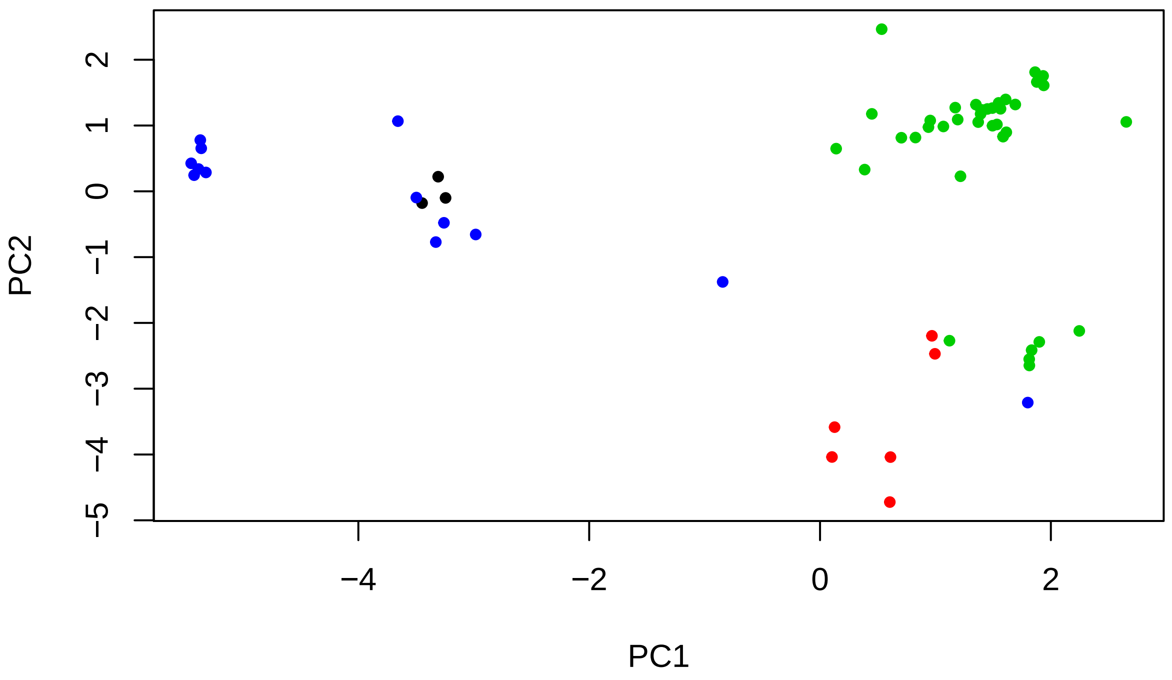


```
pairs(viruses.pca$x[,1:5],col=groups,pch=20)
```



PC2 against PC1 does indicate some separation of the classes. There are hordevirus samples, however, that seem to be grouped with the furoviruses

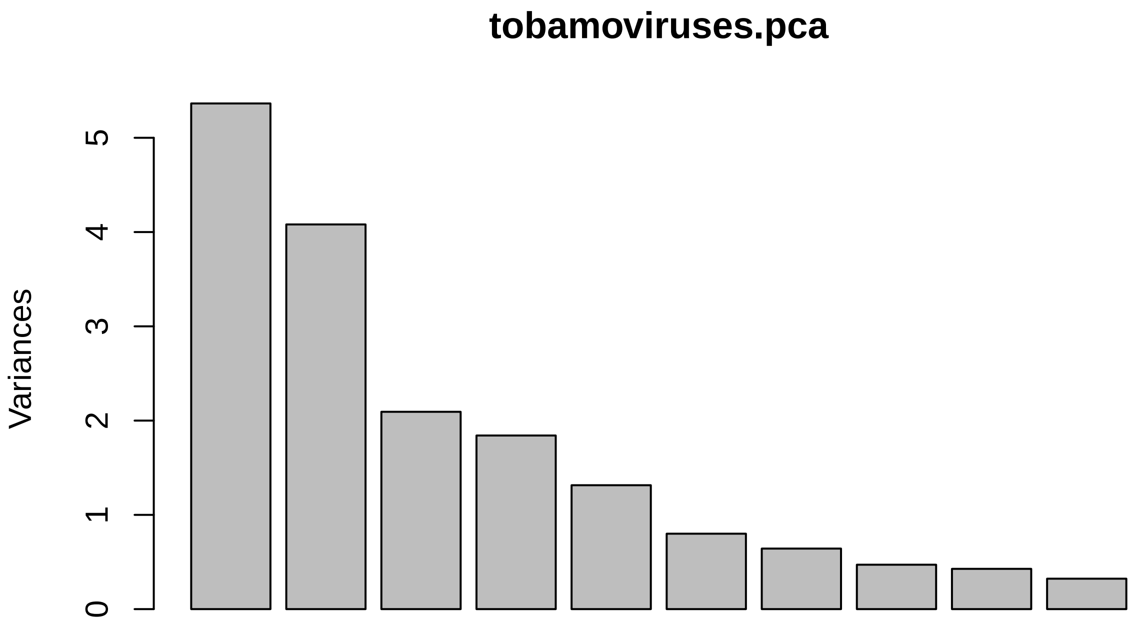
```
plot(viruses.pca$x[,1:2],pch=20,col=groups)
```



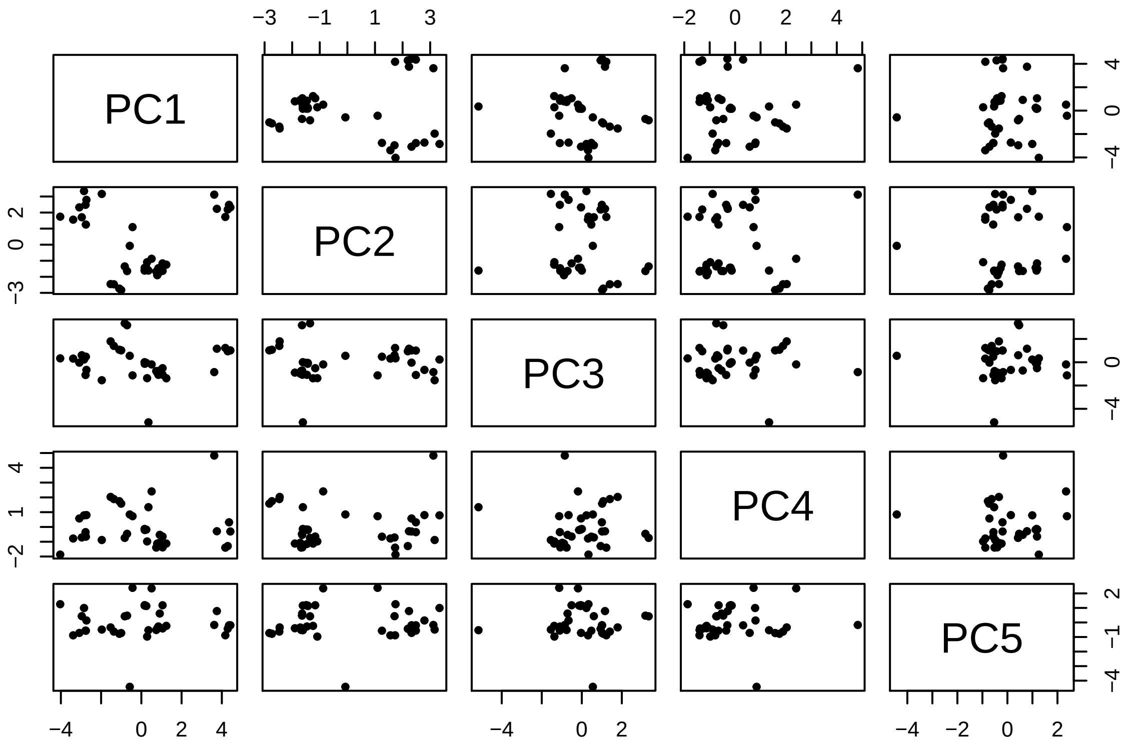
(c) The largest group of viruses is the tobamoviruses. Does a principal component analysis suggest there might be subgroups within this group of viruses?

```r
X <- scale(tobamoviruses)
tobamoviruses.pca <- prcomp(X)

plot(tobamoviruses.pca)
```



```r
pairs(tobamoviruses.pca$x[,1:5],pch=20)
```



It does appear that there might be 3 subgroups of the tobamoviruses.

```
plot(tobamoviruses.pca$x[,1:2],pch=20)
```