Graphical models and PCA (Spam Filtering)

Rob Leonard (robleonard@tamu.edu (mailto:robleonard@tamu.edu))

Load any packages you need here

Introduction

For this assignment, let's attempt to make a spam filter. Usually, this would involve a lot of text processing on a huge number of emails. In this case, someone has created a feature matrix for us. The feature matrix has rows given by individual emails and columns given by the number of each word or character that appears in that email, as well as three different numerical measures regarding capital letters (average length of consecutive capitals, longest sequence of consecutive capitals, and total number of capital letters).

Data Loading

Read in the R data set spam.Rdata (note: when the object is an Rdata file, use load('spam.Rdata') and read the documentation file spambase.Documentation.

```
load('spam.Rdata')
Xfull = spam$XdataF

#Let's remove the 'word' features
X = Xfull %>% select(contains('punc'),contains('cap'))
#How many observations are in the data set.
n = nrow(X)
#How many features are there left?
p = ncol(X)
```

We have 4601 number of emails (observations). There are 9 number of words (features).

Graphical models

2.1. The bootstrap

Estimate a partial correlation graph with the bootstrap using 500 bootstrap draws (typically, we would use more). Draw an edge between two nodes if 0 is not in the bootstrap interval. Remember: Omega_jk = 0 means the two features are partially uncorrelated given every other feature.

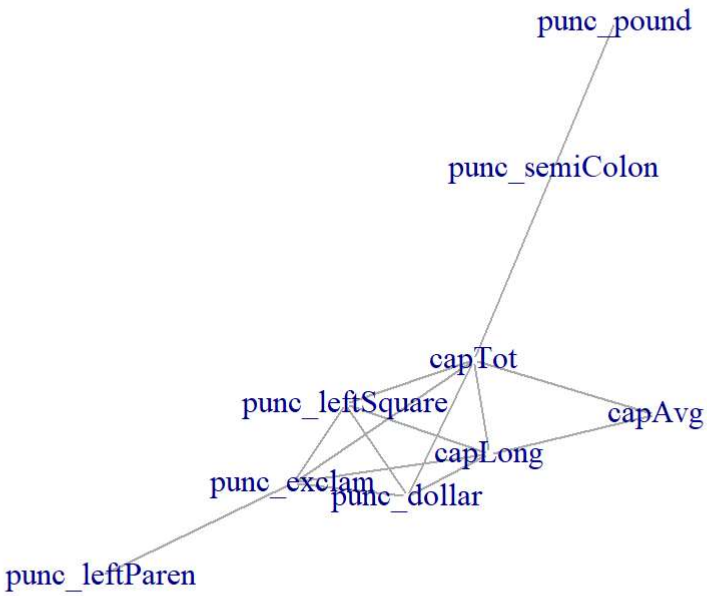
```
B = 500 # set number of bootstrap draws
Rhat_staroutput = array(dim=c(p,p,B))
for(b in 1:B) {
    bootSamp = sample(1:n,n,replace=TRUE)
    X_star = X[bootSamp,]
    S_star = cov(X_star)
    OmegaHat_star = solve(S_star)
    Rhat_star = -diag(1/sqrt(diag(OmegaHat_star))) %*% OmegaHat_star  %*% diag(1/sqrt(diag(Omega
Hat_star)))
    Rhat_staroutput[,,b] = Rhat_star
}

apply(Rhat_staroutput[1:5,1:5,],1:2, quantile, .005) # Print the lower 0.005 quantile of the boo
tstrap draws for the first 5 features only
```

```
##              [,1]         [,2]         [,3]         [,4]        [,5]
## [1,] -1.000000000 -0.013850597 -0.002863863 -0.009583336 -0.03278078
## [2,] -0.013850597 -1.000000000 -0.003556823 -0.094978213 -0.03402202
## [3,] -0.002863863 -0.003556823 -1.000000000 -0.041165234 -0.03897602
## [4,] -0.009583336 -0.094978213 -0.041165234 -1.000000000  0.06145489
## [5,] -0.032780777 -0.034022015 -0.038976016  0.061454888 -1.00000000
```

```
apply(Rhat_staroutput[1:5,1:5,],1:2, quantile, .995) # Print the upper 0.995 quantile of the boo
tstrap draws for the first 5 features only
```

```
##             [,1]        [,2]        [,3]        [,4]        [,5]
## [1,] -1.00000000  0.09523228  0.02948887  0.06626578  0.01629602
## [2,]  0.09523228 -1.00000000  0.08836797 -0.02561192  0.02882994
## [3,]  0.02948887  0.08836797 -1.00000000 -0.01788497 -0.01171221
## [4,]  0.06626578 -0.02561192 -0.01788497 -1.00000000  0.24671134
## [5,]  0.01629602  0.02882994 -0.01171221  0.24671134 -1.00000000
```

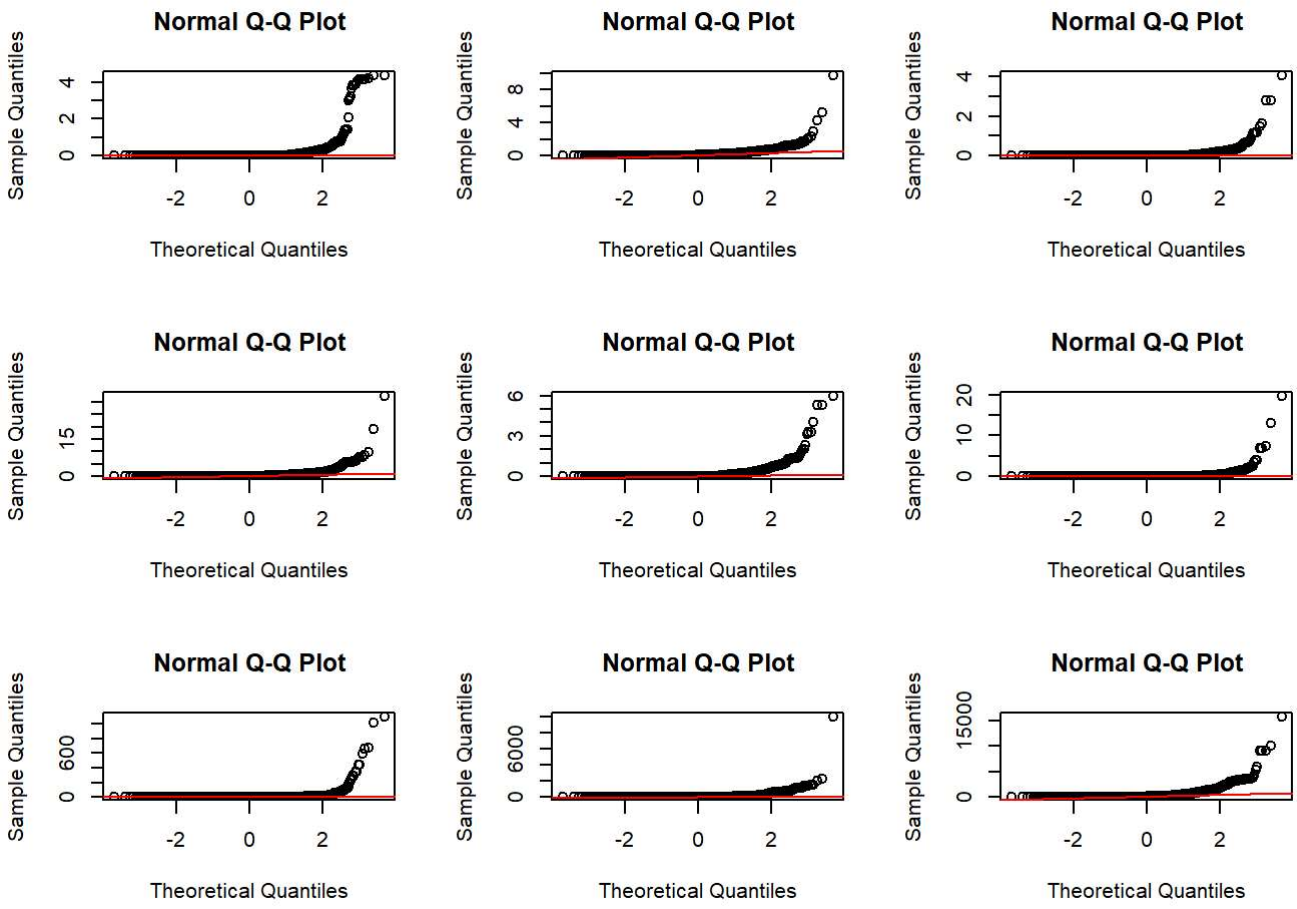
```
noEdge = 0 > apply(Rhat_staroutput,1:2, quantile, 0.005) & 0 < apply(Rhat_staroutput,1:2, quanti
le, 0.995) # produce a partial correlation graph using these bootstrap confidence intervals for
 all the non-word features.

g = graph.adjacency(!noEdge , mode="undirected", diag=FALSE) # a graph
plot(g, layout=layout.auto,
     vertex.color='white',vertex.size=3,vertex.label=names(X),vertex.frame.color=NA)
```



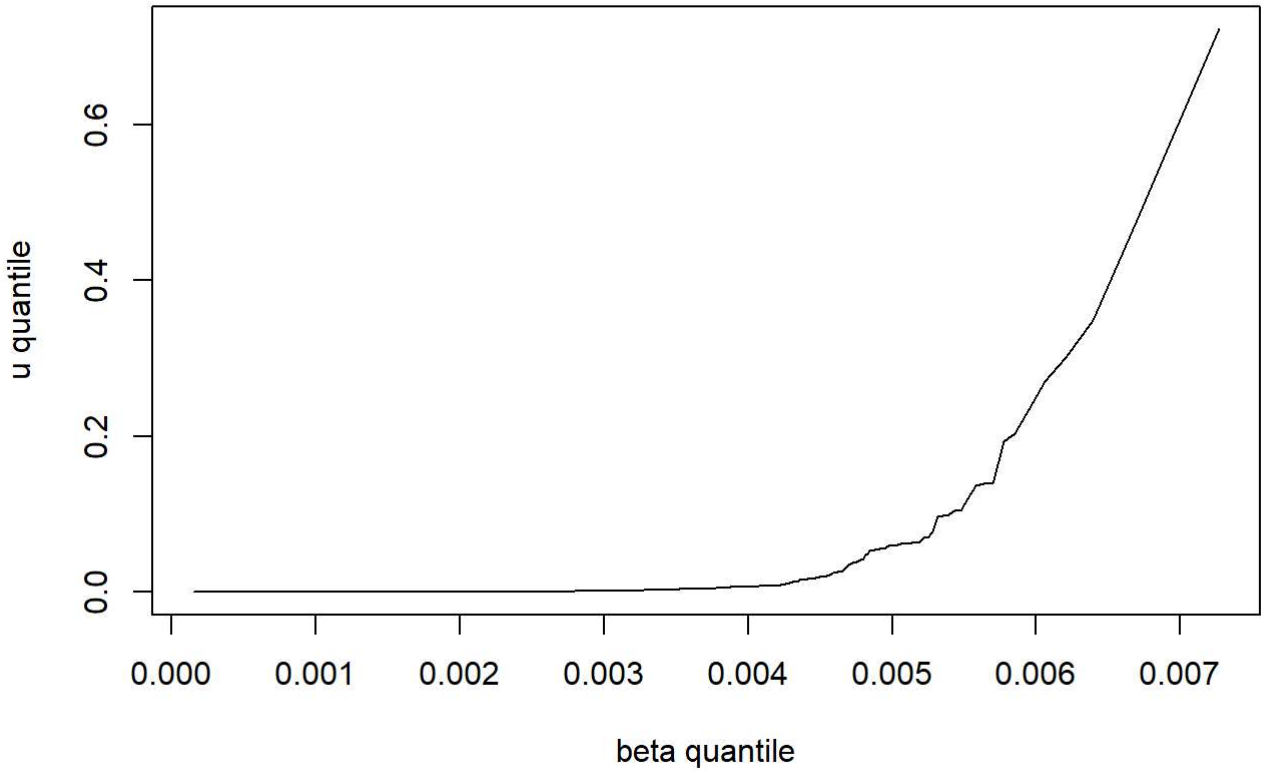
2.2. Multivariate Normal

Let's investigate if the partial correlation graph interpretation can be strengthened to conditional independence in this case.

```
# Plot the QQplots looking at some of the marginal distributions
par(mfrow= c(3,3))
for(j in 1:p){
  qqnorm(X[,j])
  qqline(X[,j],col = 'red')
}
```




```
# Get the QQplot  with respect to the beta distribution
alp = (p-2)/(2*p)
bet = (n-p-3)/(2*(n-p-1))
a   = p/2
b   = (n-p-1)/2
probs = (1:n-alp)/(n-alp-bet+1)
quantiles = qbeta(probs,a,b)     # get theoretical quantiles for x-axis
u = n*(mahalanobis(X,colMeans(X),cov(X)))/((n-1)^2) # need to calculate the Mahalanobis distance
s, in standard ui
par(mfrow=c(1,1))
plot(quantiles,sort(u),type='l',xlab='beta quantile',ylab = 'u quantile')
```



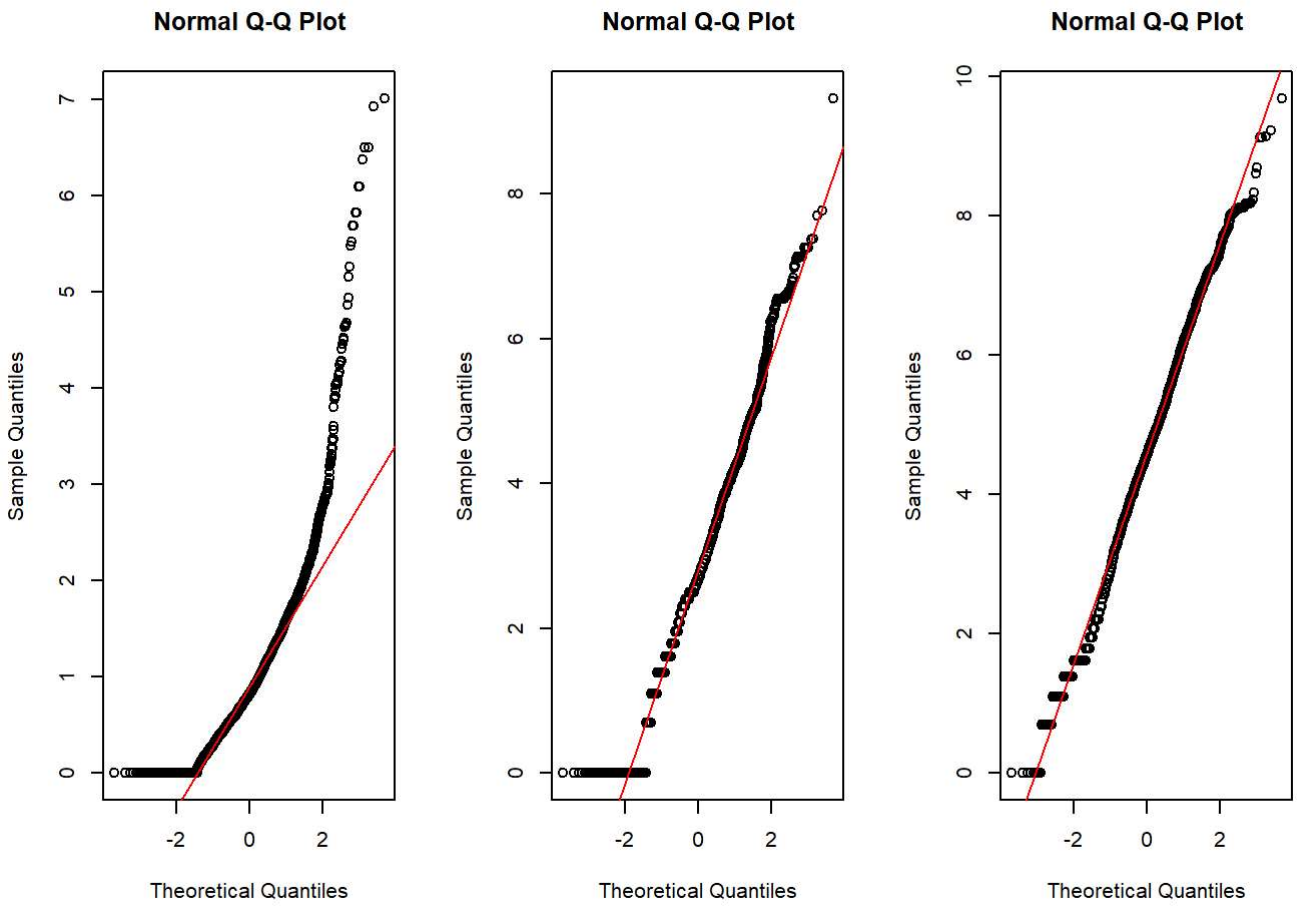
Looking at this data, is it reasonable to extend these results to conditional independence?
No, it is not reasonable, as the data is not approximately multivariate normal. None of the individual qq plots or the Mahalanobis distance qq plots are close to linear.

Write down an interpretation of this graphical model (it shouldn't be comprehensive, just interpret some aspect of the graphical model. One or two short sentences is fine).
There is a relationship (a partial correlation) between the pound and semi-colon punctuation symbols according the the graph, after removing the effects of all the other features. There does not seem to be a partial correlation between the pound and left parenthesis symbol after removing the effects of the other features.

Box-Cox Transformation to Normality

Let's instead attempt to transform our data to something more normal. For this section, let's just look at the 'capitalization' features. Write up a function for finding a the multivariate Box-cox transformation, transform the data, and then re-make the marginal qqplots.

```
X = Xfull %>% select(contains('cap'))
p = ncol(X)
Xlog = log(X)

S_lamF = function(lam, X){ # Compute the sample covariance matrix on the transformed data
  Xpow = data.frame(cbind(((X[,1]^(lam[1])-1)/(lam[1]))),((X[,2]^(lam[2])-1)/(lam[2])),((X[,3]^
(lam[3])-1)/(lam[3])))
  Xpowcen = scale(Xpow,center=TRUE,scale=FALSE)
  sigmapow = (1/(n-1))*t(Xpowcen) %*% Xpowcen  # calc cov matrix, could also use cov()
  return(list("Sigma"=sigmapow,"Xpow"=Xpow))
}

logLikeF = function(lam, X2=X){  # Compute the loglik on the transf data
  loglikout = -(n/2)*log(det(S_lamF(lam,X)$Sigma))+((lam[1]-1)*apply(Xlog[1],2,sum))+((lam[2]-1)
*apply(Xlog[2],2,sum))+((lam[3]-1)*apply(Xlog[3],2,sum))
  return(loglikout)
}

lam = c(0.0001, 0.0005, 0.001,.0025,.005,.01,.05) #these values are due to the type of skew
lamGrid = expand.grid(lam,lam,lam)
likelihoodout = data.frame(cbind(lamGrid,0))

for(i in 1:nrow(lamGrid)){
    vec1 = c(lamGrid[i,1],lamGrid[i,2],lamGrid[i,3])
    likelihoodout[i,4] = logLikeF(vec1,X)
}

testlam = which.max(likelihoodout[,4])
lambdaHat = likelihoodout[testlam,1:3] # Replace this with the argmax
# now check qq plots with transformed data
Xtransform = data.frame(cbind(cbind(((X[,1]^(.0001)-1)/(.0001))),((X[,2]^(.0025)-1)/(.0025)),((X
[,3]^(.0001)-1)/(.0001))))
par(mfrow= c(1,3))
for(j in 1:p){
  qqnorm(Xtransform[,j])
  qqline(Xtransform[,j],col = 'red')
}
```



I found the transformation parameter lambda to be 10^{-4}, 0.0025, 10^{-4}. The transformed data still does not look multivariate normal as plots are well off the normal line.

Principal Component Analysis

Let's look at the 'stock_prices.csv' data. These contain normalized stock prices for five different companies over a period of weeks, three of which are banks (JP Morgan, Citibank, and Wells Fargo) and two of which are oil companies (Shell and Exxon Mobil).

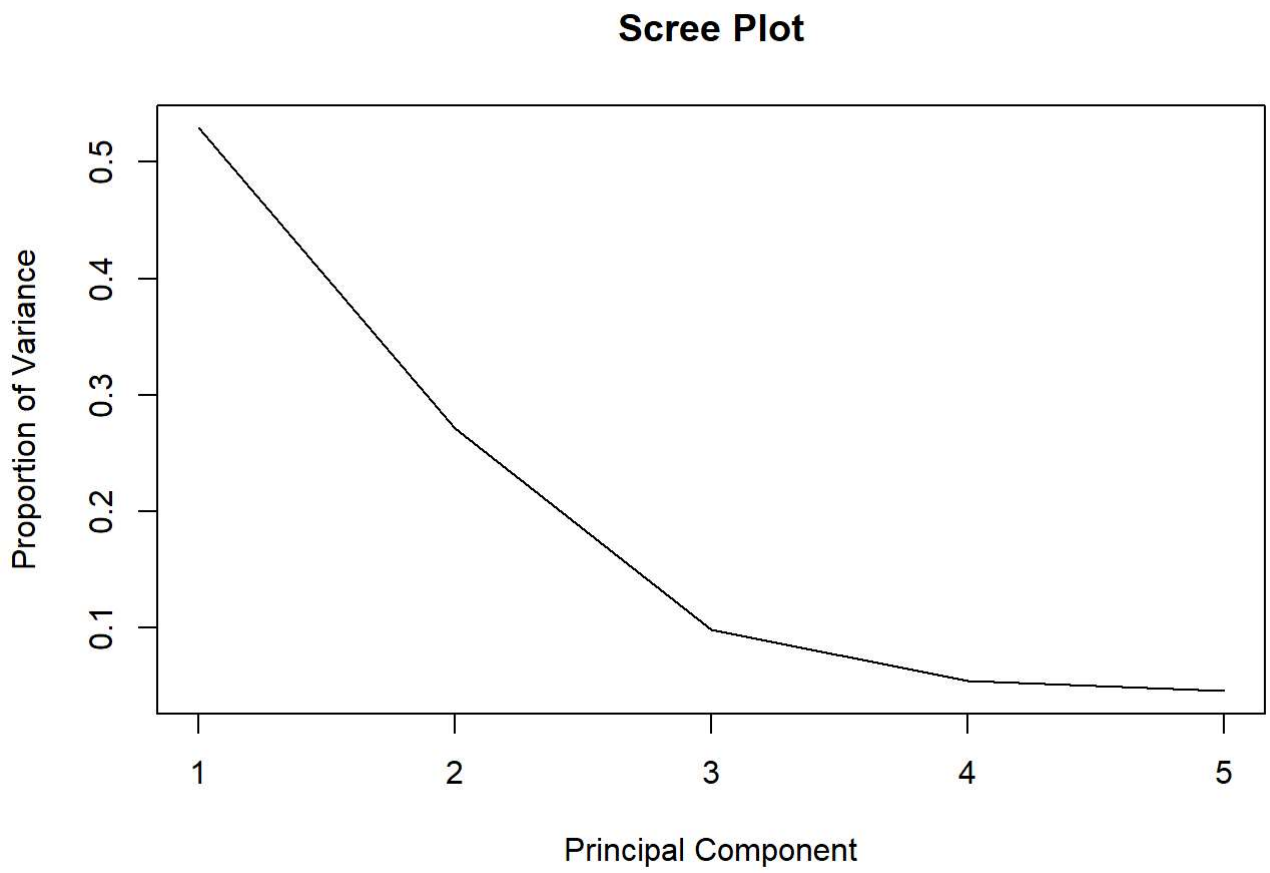
3.1. Scree Plots

Get the PCA of this data, but only center by the mean, don't scale by the standard deviation. Make a scree plot of the proportion of variance explained against principal component (PC) number. How many principal components would you recommend in order to retain 90% of the variance?

```
stockdata = read.delim("stock_prices.csv", header=TRUE, sep=",")
out = prcomp(stockdata,center=TRUE,scale=FALSE)
pcs = out$rot
scores = out$x
min(which(cumsum(out$sdev**2/sum(out$sdev**2))>.9))
```

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

```
yvec = (out$sdev**2/sum(out$sdev**2))
xvec= c(1,2,3,4,5)
plot(xvec,yvec, type="l",xlab="Principal Component",ylab="Proportion of Variance",main="Scree Pl
ot") # scree plot - proportion of var
```



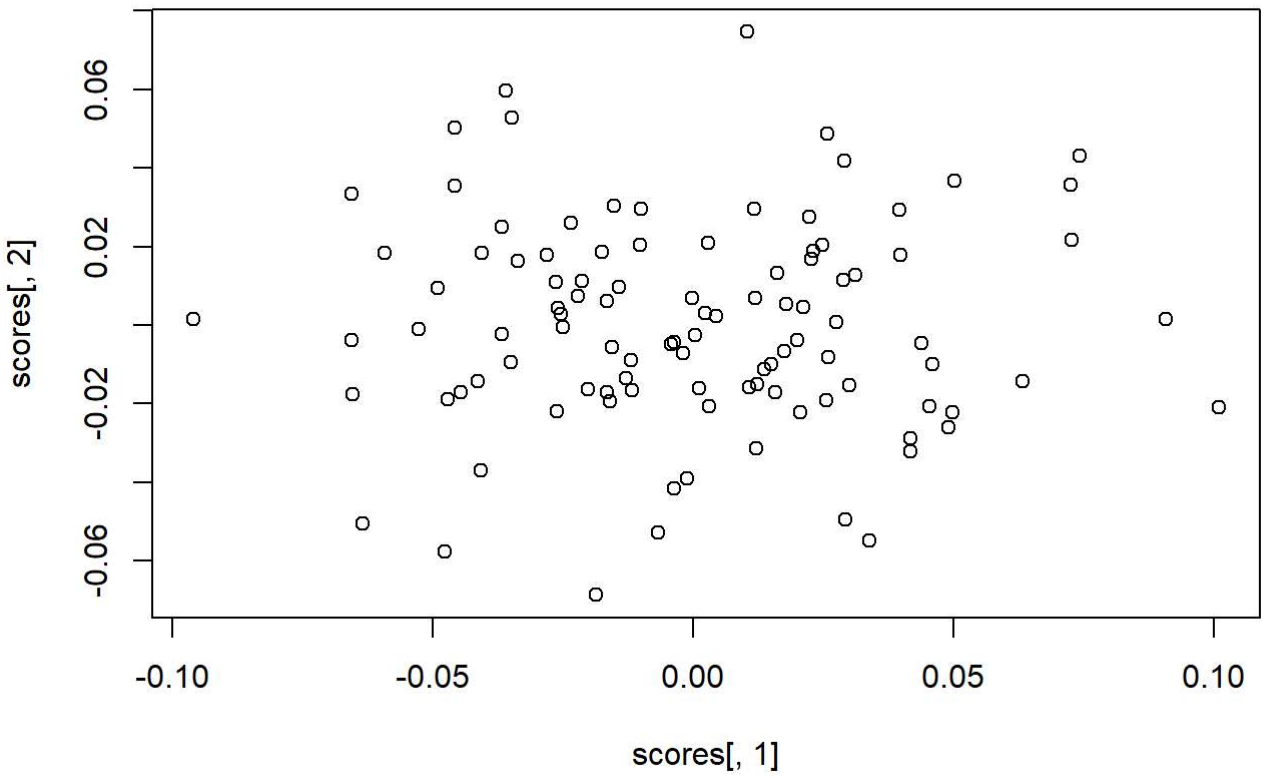

```
nPCs = 4 # number of retained PCs
```

I recommend retaining 4 PCs.

3.2. PCA for exploratory data analysis

Make a scatterplot of the first two principal component scores. Which weeks have the minimum and maximum values of PC 1?

```
plot(scores[,1],scores[,2]) # scatter plot
```




```
(weekWithMinPC1score = which.min(scores[,1]))
```

```
## [1] 56
```

```
(weekWithMaxPC1score = which.max(scores[,1]))
```

```
## [1] 63
```

The week with the minimum PC1 score is the observation 56 and the maximum PC1 score is 63.

Comment on how these weeks compare to each other. In week 56 (min PC1) all stock returns are positive, with the oil companies having greater returns than the banks. In week 63 (max PC1) all stock returns are negative, again with oil companies having greater price reductions than the banks. So the min and max are moving/pulling in opposite directions.

3.3. To scale or not to scale?

Now run PCA again but centering and scaling the data. Find the percent variance explained by the first ‘nPCs’ for the unscaled PCA and for the scaled PCA.

```
out2 = prcomp(stockdata,center=TRUE,scale=TRUE)
pcs2 = out2$rot
scores2 = out2$x
min(which(cumsum(out2$sdev**2/sum(out2$sdev**2))>.9))
```

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

```
(percVarExplainedUnscaled = sum(out$sdev[1:4]**2/sum(out$sdev**2)))*100
```

```
## [1] 95.39935
```

```
(percVarExplainedScaled   = sum(out2$sdev[1:4]**2/sum(out2$sdev**2)))*100
```

```
## [1] 94.8966
```

The percent var explained unscaled is 0.9539935 and the percent var explained scaled is 0.948966

Is there much of a difference between the scaled and unscaled PCA for this data set? Why or why not?
There isn't much of a difference as the returns are measured in the same units (percentages) and the returns are already normalized as well.