Chemometrics Midterm

Katie Daisey

Wednesday, April 15, 2015

library(knitr)  
opts\_chunk$set(fig.pos="center")

1a - assign spectra? 1c - offset? 1e

## Question 1 - Multivariate Calibration

Here we wish to examine spectral data of several mixtures with 4 components (acetone, i-propyl alcholo, water, and t-butyl alcohol). The data also consists of a calibration set of 24 mixtures and a prediction set of 12 mixtures.

data<-read.csv("data/shootout2.csv")  
cal<-data[1:24,]  
pred<-data[25:36,]

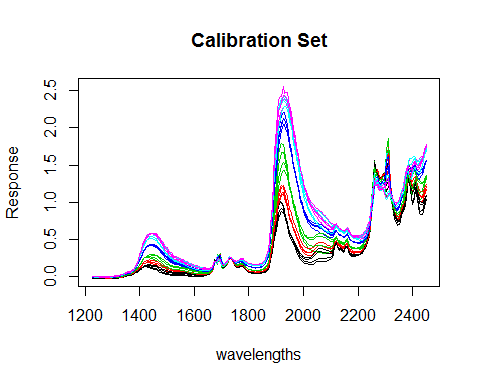
There is a known mistake in the acetone concentration of sample 13. It should read 60, not 6.

cal[13,248]<-60

#### a) CLS Regression

We wish to do a simple classical least-squares regression on the calibration data. First we must separate the spectral information from the mixture composition. We can also take a quick look at the spectra themselves.

calspec<-as.matrix(cal[,2:247],ncol=246) #conversion necessary for future code  
calconc<-as.matrix(cal[,248:251],ncol=4) #conversion necessary for future code  
  
wavelengths<-seq(1225,2450,length.out=246)  
matplot(wavelengths,t(calspec),type="l", main="Calibration Set", ylab="Response", lty=1)

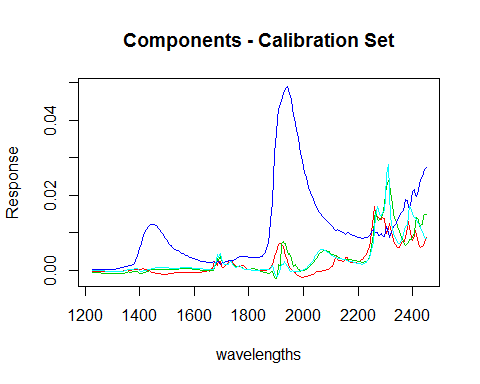


We use matrix algebra () to calculate the coefficients for each variable (here - component) and multiply that by the spectra to extract the spectra of the pure components.

rcond(t(calconc)%\*%calconc)

## [1] 0.0007182945

kmat<-solve(t(calconc)%\*%calconc)%\*%t(calconc)%\*%calspec  
matplot(wavelengths,t(kmat),type="l", main="Components - Calibration Set", ylab="Response", lty=1, col=2:5)

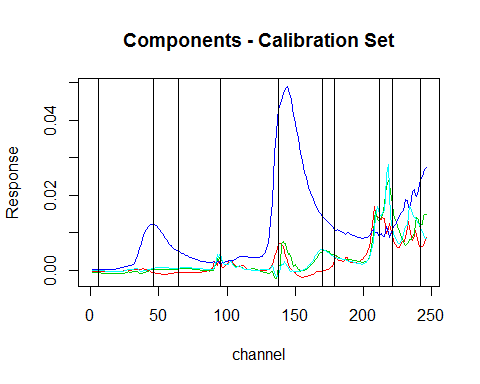


First, we see a low reciprocal condition number which means the regular matrix is ill-conditioned and suspectible to small changes in values. However, the reciprocal number could be much lower, and this is the only data we have, we should just remember our predicted concentrations may not be very precise. We might also consider trying indirect least squares.

#### b) Selection of peaks for ILS

ILS requires variables less than or equal to the rank of the matrix (the smaller of n and p in an n x p matrix). We select 10 wavelengths which we beleive to best separate the four components.

pick<-c(6,46,64,95,138,170,179,212,221,242)  
matplot(1:246,t(kmat),type="l", main="Components - Calibration Set", ylab="Response",xlab="channel", lty=1, col=2:5)  
abline(v=pick)



calpick<-calspec[,pick]  
rcond(t(calpick)%\*%calpick)

## [1] 1.05572e-08

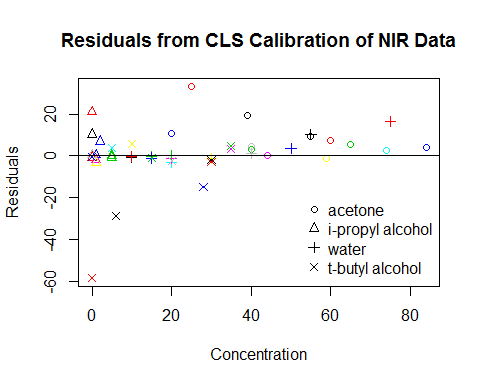
#### c) Offset for CLS

Using Classical Least Squares we can calculate the square root of the sum of the residuals between our predicted concentrations and our actual concentrations for each component.

rcond(kmat%\*%t(kmat))

## [1] 0.006561728

predspec<-as.matrix(pred[,2:247],ncol=246)  
predconc<-as.matrix(pred[,248:251],ncol=4)  
kmat\_predconc<- predspec%\*%t(kmat)%\*%solve(kmat%\*%t(kmat))  
  
rownames(predconc)<-1:12  
plot(predconc,predconc-kmat\_predconc,xlab="Concentration",ylab="Residuals",  
 main="Residuals from CLS Calibration of NIR Data",  
 pch=rep(1:4,each=12), col=rownames(predconc))  
abline(h=0)  
legend("bottomright", c("acetone", "i-propyl alcohol", "water", "t-butyl alcohol"), pch=1:4, bty='n')



Here we see a plot of the residuals versus actual concentration with each sample having a different color and each component sharing a symbol

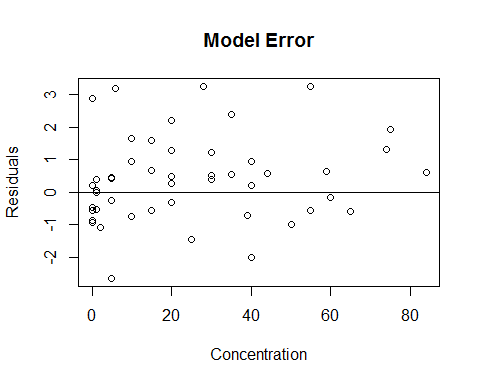
(cls\_rmsep<-sqrt(apply(((predconc-kmat\_predconc)^2),2,mean))) # RMSEP

## Acetone IPA Water TBA   
## 12.262377 7.136933 5.816392 19.489254

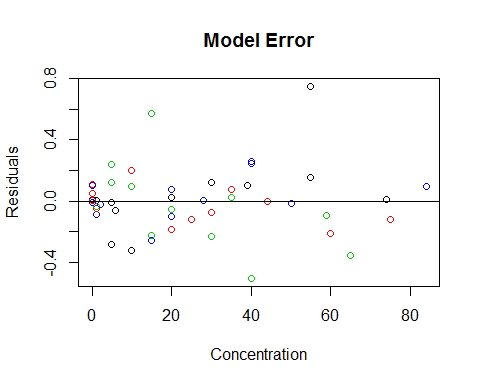
The RMSEPs for our classical least-squares model are 12.26238, 7.13693, 5.81639, 19.48925 respectively.

#### d) ILS

predpick<-predspec[,pick]  
  
#create model  
kmatils<-solve(t(calpick)%\*%calpick)%\*%t(calpick)%\*%calconc  
  
#project unknowns into model  
predils<-predpick%\*%kmatils  
  
#plot residuals from known concentrations  
plot(predconc,predconc-predils, main="Model Error", xlab="Concentration", ylab="Residuals")  
abline(h=0)



pred<-matrix(0,nrow=12,ncol=4)  
for(i in 1:4){  
a<-solve(t(predpick)%\*%predpick)%\*%t(predpick)%\*%predconc[,i]  
pred[,i]<-predpick%\*%a  
}  
colnames(pred)<-1:4  
plot(predconc,predconc-pred,col=colnames(pred),main="Model Error", xlab="Concentration", ylab="Residuals")  
abline(h=0)



#### e) CLS versus ILS

(ils\_rmsep<-sqrt(apply((predconc-pred)^2,2,mean)))

## Acetone IPA Water TBA   
## 0.30329186 0.12033532 0.24708376 0.09691661

cls\_rmsep

## Acetone IPA Water TBA   
## 12.262377 7.136933 5.816392 19.489254

Comparing the root mean squared residuals we would say that CLS preforms better than ILS, however ILS is much less intensive to run.

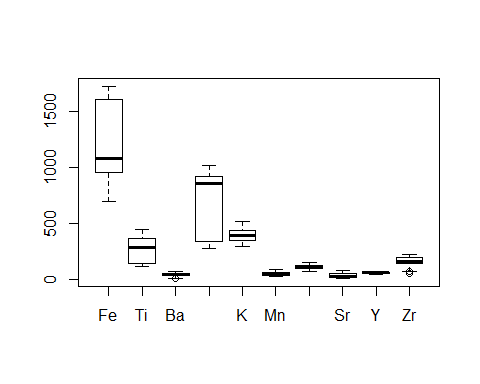
## Question 2 - Provenance of Archaeological Artifiacts

Here we are exploring obsidian arrowheads believed to be mined from four possible quarries in Mexico. 10 elemental components of 63 arrowheads of known origin were measured using neutron activation analysis. Additionally, 12 arrowheads of unknown origin were also measured.

data<-read.csv("data/arrows.csv")  
str(data)

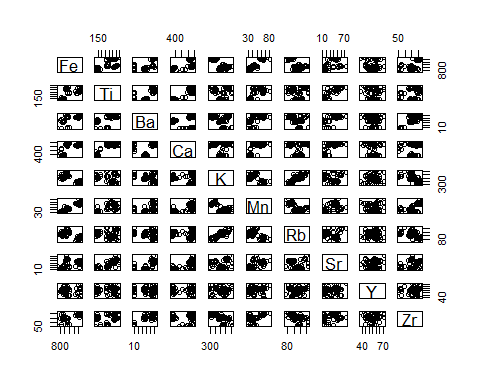
## 'data.frame': 75 obs. of 13 variables:  
## $ SampleID: Factor w/ 59 levels "ANA-1","ANA-2",..: 28 20 23 24 21 22 25 27 26 19 ...  
## $ Class : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ Quarry : Factor w/ 5 levels "AN","BL","K",..: 3 3 3 3 3 3 3 3 3 3 ...  
## $ Fe : int 1100 1173 1164 1030 1077 1080 1020 1050 1100 1069 ...  
## $ Ti : int 390 417 404 373 373 403 360 396 373 375 ...  
## $ Ba : int 55 54 56 59 55 53 59 56 53 51 ...  
## $ Ca : int 920 961 916 920 888 919 883 924 910 958 ...  
## $ K : int 460 441 446 487 455 442 473 482 477 429 ...  
## $ Mn : int 45 47 42 38 38 41 43 48 51 42 ...  
## $ Rb : int 120 135 120 128 97 133 119 140 137 100 ...  
## $ Sr : int 57 55 58 53 51 60 40 74 61 51 ...  
## $ Y : int 58 60 45 58 54 45 50 71 58 47 ...  
## $ Zr : int 142 145 148 138 145 155 134 157 152 128 ...

data2<-data  
  
#12 unknowns located last  
ukn<-data[64:75,]  
data<-data[-(64:75),]  
  
#property information (sampleID, Class, Quarry) in first 3 columns  
prop<-data[,1:3]  
data<-as.matrix(data[,-(1:3)],ncol=10)  
ukn\_prop<-ukn[,1:3]  
ukn\_data<-as.matrix(ukn[,-(1:3)],nrow=12)  
  
par(mfrow=c(1,1))  
boxplot(data)



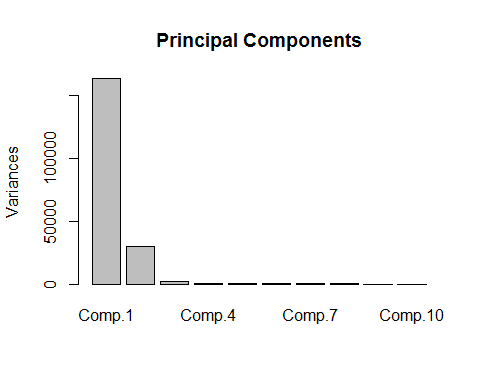
A boxplot of the data shows a wide spread in the values for the different elements as well as vastly differing variances. Auto-scaling this data will probably be necessary.

plot(as.data.frame(data))



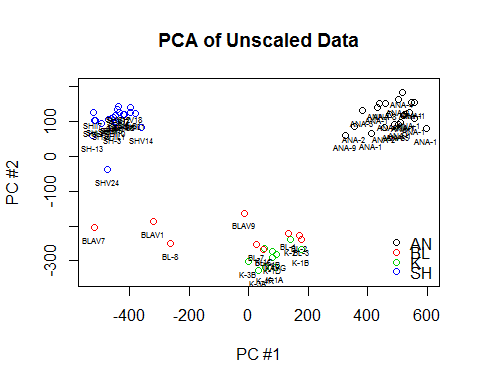
Plotting the elements against each other show several loose correlations, for instance between K and Rb suggesting there may be replicate information. We also see several distinct groups separated by only two elements.

pca<-princomp(data)  
plot(pca, main="Principal Components")

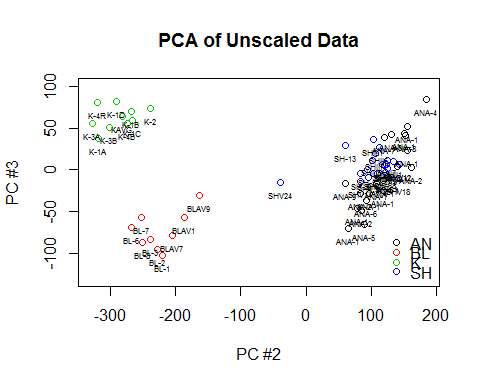


Doing a quick pca on the data, we find 3 principal components.

plot(pca$scores[,1],pca$scores[,2],col=prop[,3],ylim=c(-350,200),  
 xlab="PC #1", ylab="PC #2", main="PCA of Unscaled Data")  
text(pca$scores[,1],pca$scores[,2],labels=prop[,1],cex=.5,pos=1)  
legend("bottomright",col=1:4,pch=1,c("AN", "BL", "K", "SH"),bty='n',y.intersp=.5)

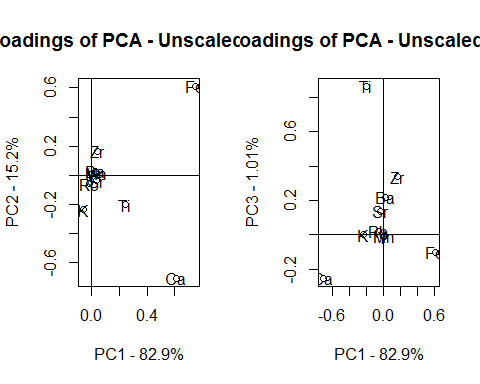


plot(pca$scores[,2],pca$scores[,3],col=prop[,3],ylim=c(-130,100),  
 xlab="PC #2", ylab="PC #3", main="PCA of Unscaled Data")  
text(pca$scores[,2],pca$scores[,3],labels=prop[,1],cex=.5,pos=1)  
legend("bottomright",col=1:4,pch=1,c("AN", "BL", "K", "SH"),bty='n',y.intersp=.5)



The first two components creates tight clusters for AN, SH, and K samples (with the exception of SHV24), however the BL samples are more widespread and do not separate from the K samples. The K and BL samples do separate cleanly with the addition of the third principal component.

par(mfrow=c(1,2))  
plot(pca$loadings[,1],pca$loadings[,2], xlab="PC1 - 82.9%", ylab="PC2 - 15.2%", main="Loadings of PCA - Unscaled Data")  
text(pca$loadings[,1],pca$loadings[,2],labels=colnames(data))  
abline(h=0,v=0)  
  
plot(pca$loadings[,2],pca$loadings[,3], xlab="PC1 - 82.9%", ylab="PC3 - 1.01%", main="Loadings of PCA - Unscaled Data")  
text(pca$loadings[,2],pca$loadings[,3],labels=colnames(data))  
abline(h=0,v=0)

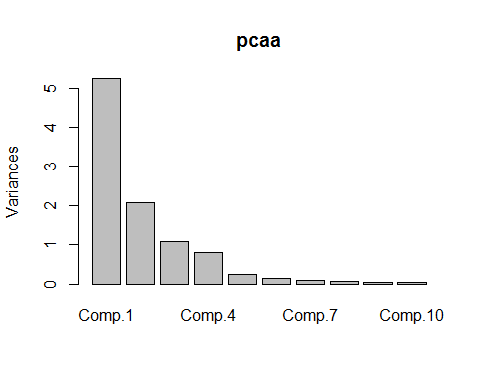


We see that Fe and Ca contribute the most to PC1 which contains the vast majority of the variance. Ti contributes the most to PC3 which is necessary to separate the BL and K groups.

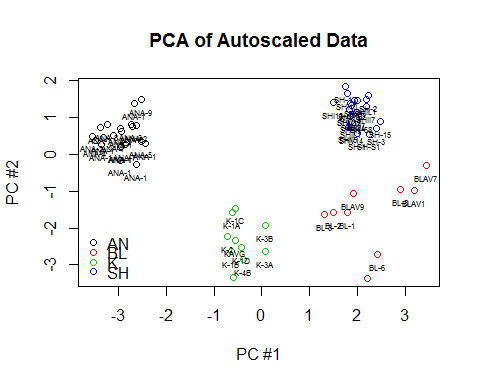
#### b) Auto-scaled Data

As mentioned after the boxplot, we may wish to autoscale the data.

data\_a<-scale(data)  
center<-attributes(data\_a)[[3]]  
scale<-attributes(data\_a)[[4]]  
pcaa<-princomp(data\_a)  
par(mfrow=c(1,1))  
plot(pcaa)

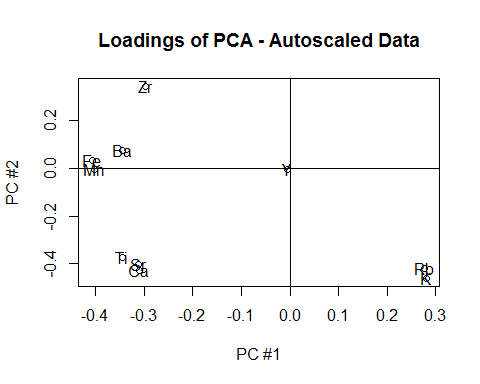


plot(pcaa$scores[,1],pcaa$scores[,2],col=prop[,3],  
 xlab="PC #1", ylab="PC #2", main="PCA of Autoscaled Data")  
text(pcaa$scores[,1],pcaa$scores[,2],labels=prop[,1],cex=.5,pos=1)  
legend("bottomleft",col=1:4,pch=1,c("AN", "BL", "K", "SH"),bty='n',y.intersp=.5)



With the auto-scaled data we see clear separation of all the groups with only the first two components. We still see a wide spread in the BL group.

plot(pcaa$loadings[,1],pcaa$loadings[,2],  
 xlab="PC #1", ylab="PC #2", main="Loadings of PCA - Autoscaled Data")  
abline(v=0,h=0)  
text(pcaa$loadings[,1],pcaa$loadings[,2],labels=colnames(data\_a))

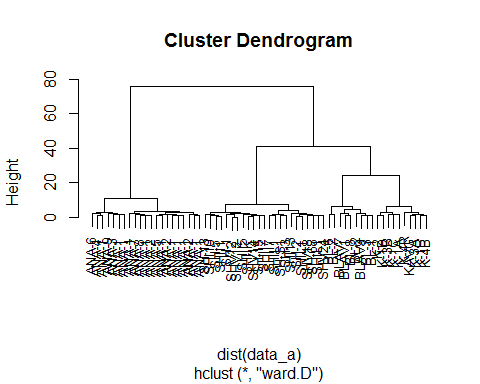


Now Fe seems to play a smaller role, while several elements (Zr, Rb, K, Ca, Sr, Ti) have a large effect on the principal components. It is likely some of these elements contain replicated data, especially considering the strong correlation we saw earlier between Rb and K.

1. Hierarchical Clustering

We also wish to hierarchically cluster the arrows to see if four distinct groups form.

hc1<-hclust(dist(data\_a),"ward.D")  
plot(hc1, labels=prop[,1],cex=.8)



With auto-scaled data and using Ward's method we do see clean separation of the quarries.

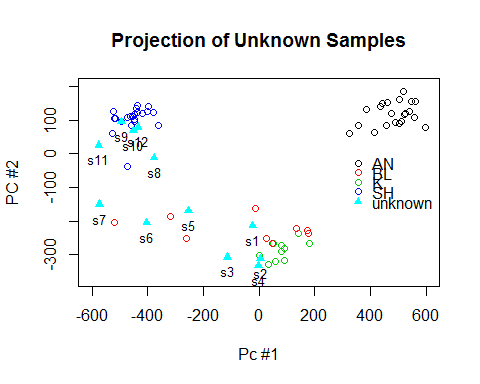
#### d) Projection of unknown samples into PCA and Clustering

Now that we have created models which separate our known samples, we wish to project our unknown samples into these models.

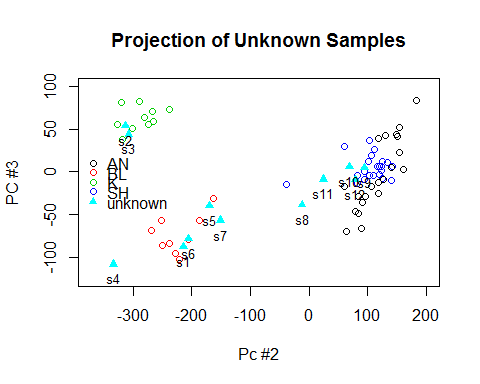
##### 1)PCA - unscaled

Since the original PCA did not scale or center the data, the unknown samples can be projected simply by multiplying by the loadings (and centering with the center from the known data pca).

test<-scale(data%\*%pca$loadings,scale=F)  
ukn\_pca<-ukn\_data%\*%pca$loadings  
ukn\_pca<-scale(ukn\_pca,center=attributes(test)$'scaled:center',scale=F)  
  
plot(pca$scores[,1],pca$scores[,2],col=prop[,3],xlim=c(-600,600),ylim=c(-370,200),  
 xlab="Pc #1", ylab="PC #2", main="Projection of Unknown Samples")  
points(ukn\_pca[,1],ukn\_pca[,2],col=5,pch=17)  
text(ukn\_pca[,1],ukn\_pca[,2],pch=17, labels=ukn\_prop[,1],cex=.8,pos=1)  
legend("right",col=1:5,pch=c(1,1,1,1,17),c("AN", "BL", "K", "SH","unknown"),bty='n',y.intersp=.5)



plot(pca$scores[,2],pca$scores[,3],col=prop[,3],xlim=c(-370,200),ylim=c(-125,100),  
 xlab="Pc #2", ylab="PC #3", main="Projection of Unknown Samples")  
points(ukn\_pca[,2],ukn\_pca[,3],col=5,pch=17)  
text(ukn\_pca[,2],ukn\_pca[,3],pch=17, labels=ukn\_prop[,1],cex=.8,pos=1)  
legend("left",col=1:5,pch=c(1,1,1,1,17),c("AN", "BL", "K", "SH","unknown"),bty='n',y.intersp=.5)



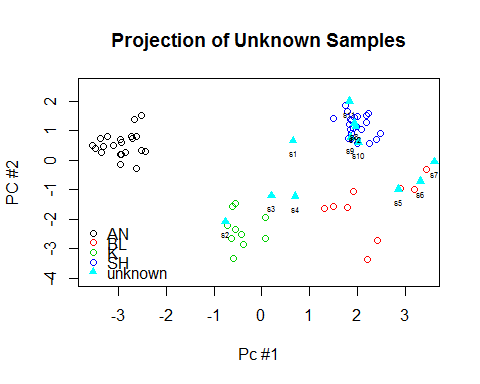
With the unscaled data we find 5 samples (s8,s9,s10,s11,s12) which would be assigned to the SH quarry. Using PC3 we would separate 2 samples (s2,s3) as from the K quarry and 5 samples (s1,s4,s5,s6,s7) as from the BL quarry.

assignpca<-c("BL","K","K","BL","BL","BL","BL","SH","SH","SH","SH","SH")

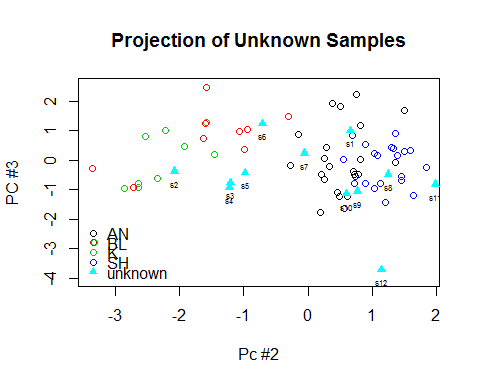
##### 2.) Auto-scaled PCA

We should hopefully observe the same assignments using autoscaled data.

ukn\_data\_a<-scale(ukn\_data,center,scale)  
ukn\_pcaa<-ukn\_data\_a%\*%pcaa$loadings  
  
plot(pcaa$scores[,1],pcaa$scores[,2],col=prop[,3],ylim=c(-4,2.5),  
 xlab="Pc #1", ylab="PC #2", main="Projection of Unknown Samples")  
points(ukn\_pcaa[,1],ukn\_pcaa[,2],col=5,pch=17)  
text(ukn\_pcaa[,1],ukn\_pcaa[,2],labels=ukn\_prop[,1],cex=.5,pos=1)  
legend("bottomleft",col=1:5,pch=c(1,1,1,1,17),c("AN", "BL", "K", "SH","unknown"),bty='n',y.intersp=.5)



plot(pcaa$scores[,2],pcaa$scores[,3],col=prop[,3],ylim=c(-4,2.5),  
 xlab="Pc #2", ylab="PC #3", main="Projection of Unknown Samples")  
points(ukn\_pcaa[,2],ukn\_pcaa[,3],col=5,pch=17)  
text(ukn\_pcaa[,2],ukn\_pcaa[,3],labels=ukn\_prop[,1],cex=.5,pos=1)  
legend("bottomleft",col=1:5,pch=c(1,1,1,1,17),c("AN", "BL", "K", "SH","unknown"),bty='n',y.intersp=.5)



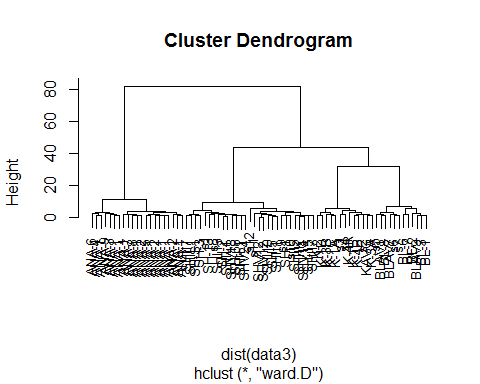
We would again assign 5 samples to SH with the exception of (s8, s9, s10, s11, and s12). 3 samples (s5, s6, and s7) would be assigned to BL while only 1 sample (s2) would clearly be assigned to K. 3 samples would not be clearly assigned (s1 - likely SH, s3 and s4 - either K or BL).

assignpcaa<-c("u","K","u","u","BL","BL","BL","SH","SH","SH","SH","SH")

No samples have been assigned to different groups than the unscaled data, but several points have not been assigned at all.

##### 3.) Hierarchical Clustering

par(mfrow=c(1,1))  
data3<-rbind(data,ukn\_data)  
data3<-scale(data3,center,scale)  
prop3<-rbind(prop,ukn\_prop)  
h2<-hclust(dist(data3),"ward.D")  
plot(h2, label=prop3[,1],cex=.8)



assignhc<-c("SH","K","K","K","BL","BL","BL","SH","SH","SH","SH","SH")  
assign<-cbind(1:12,assignpca,assignpcaa,assignhc)  
colnames(assign)<-c("Sample ","PCA ","Auto-scaled\_PCA ","Hierarchical\_clustering ")  
kable(assign)

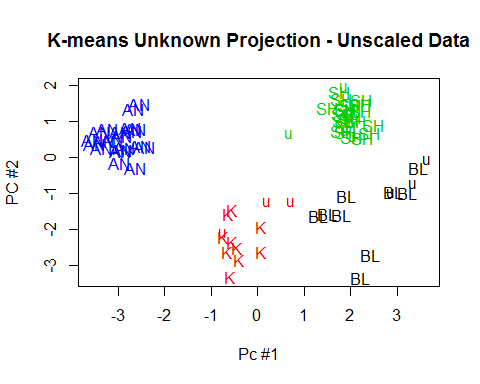
|  |  |  |  |
| --- | --- | --- | --- |
| Sample | PCA | Auto-scaled\_PCA | Hierarchical\_clustering |
| 1 | BL | u | SH |
| 2 | K | K | K |
| 3 | K | u | K |
| 4 | BL | u | K |
| 5 | BL | BL | BL |
| 6 | BL | BL | BL |
| 7 | BL | BL | BL |
| 8 | SH | SH | SH |
| 9 | SH | SH | SH |
| 10 | SH | SH | SH |
| 11 | SH | SH | SH |
| 12 | SH | SH | SH |

Based on our mis-match between our methods, we would be wary of assigning samples 1 and 4 definitively.

#### e) K-means Clustering

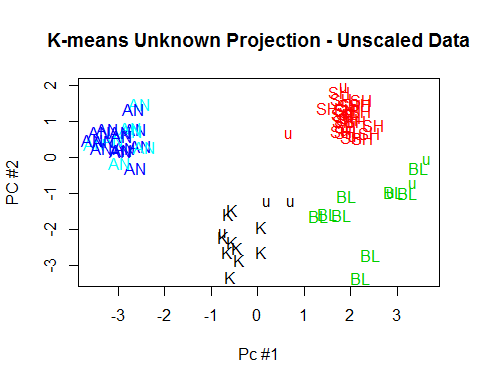
While we can visually separate the groups, are we also able to do so mathematically using k-means?

kma<-kmeans(data3,4,nstart=50)  
scores<-rbind(pcaa$scores,ukn\_pcaa)  
  
plot(scores[,1],scores[,2],col=0, xlab="Pc #1", ylab="PC #2", main="K-means Unknown Projection - Unscaled Data")  
text(scores[,1],scores[,2],col=kma$cluster,labels=prop3[,3])



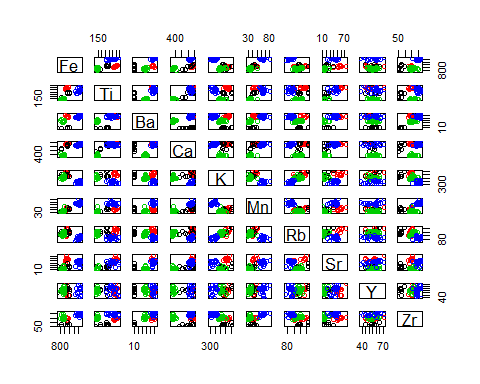
Using the auto-scaled data and searching for the expected 4 groups, we find perfect agreement between the known quarry and assign group by k-means.

kma2<-kmeans(data3,5,nstart=50)  
plot(scores[,1],scores[,2],col=0, xlab="Pc #1", ylab="PC #2", main="K-means Unknown Projection - Unscaled Data")  
text(scores[,1],scores[,2],labels=prop3[,3],col=kma2$cluster)



Interestingly, searching for 5 groups splits the AN group rather than the BL group. This is also seen in the hierarchical clustering. Since it is based on the auto-scaled data, it agrees with the auto-scaled pca and hierarchical clustering. It less closely agrees with the unscaled data .

pairs(data,col=kma$cluster)



We can once again examine all the data but with the points colored according to groups.