Lab 3 Matrix Algebra and Ordination Part I

# Set up R session

## Download packages

Install and load packages

library(MVA)  
library(MVN)  
library(psych)  
library(Hmisc)  
library(vegan)  
library(StatMatch)  
library(MASS)  
library(dplyr)

# A primer of matrix algebra

Let’s start by making our own matrix:

newMatrix<- matrix(c(1,4,5,4,5,6,9,1,9),nrow=3, ncol=3)  
newMatrix

## [,1] [,2] [,3]  
## [1,] 1 4 9  
## [2,] 4 5 1  
## [3,] 5 6 9

The command c concatenates a list of numbers.

Now let’s check the dimensions of newMatrix:

dim(newMatrix)

## [1] 3 3

## Matrix addition and subtraction

## **Question 1:** Create a new matrix to either add to or subtract from “newMatrix.” This new matrix should be a 3 x 3 matrix containing all ones and call it oneMatrix. (20 pts)

oneMatrix <- matrix(c(5,7,3,5,4,6,8,2,1), nrow = 3, ncol = 3)  
oneMatrix

## [,1] [,2] [,3]  
## [1,] 5 5 8  
## [2,] 7 4 2  
## [3,] 3 6 1

Now add oneMatrix to newMatrix:

newMatrix + oneMatrix #each cell in newM is added to the corresponding cell in oneM

Then subtract oneMatrix from newMatrix:

newMatrix - oneMatrix

**Remember, because matrix addition and subtraction is performed on an element by element basis, matrices must have the same dimensions.**

## Scalar Multiplication

A **Scalar** is a single number. Scalar multiplication multiplies a scalar times a matrix:

3\*newMatrix

An *eigenvalue* is a scalar that is an essential component of multivariate analysis. We will explore this in a little bit.

## Matrix Multiplication

You use % to signify that are using a matrix operation. Otherwise, R will attempt the operation element by element.

oneMatrix%\*%newMatrix

order matters:

newMatrix%\*%oneMatrix

**The number of columns in the first matrix must equal the number of rows in the second matrix.**

## Matrix transposition

**Transposing** a matrix involves interchanging its rows and columns:

transMatrix<-t(newMatrix)

## Identity Matrices

An **identity matrix** is a matrix where all the diagonal terms equal one and the remaining elements equal 0:

Identity<-diag(3)

## Matrix Inversion

The inverse of matrix A is A-1.

invMatrix<-solve(newMatrix)

Multiplying a matrix by its inverse yields and identity matrix (A x A-1 = I):

invMatrix%\*%newMatrix

Let’s round it:

round(invMatrix%\*%newMatrix,10)

## Eigenvalues and eigenvectors

Remember that an eigenvalue is a special scalar and the associated eigenvector is a vector that are key components of PCA.

eig<-eigen(newMatrix)

# Principal Component Analysis (PCA)

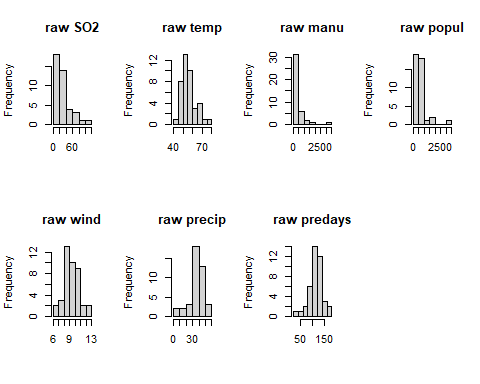
You are going to conduct a PCA on the USairpollution data that we used in Lab 2.

First let’s call in the data:

usAir<-USairpollution

Now, look at the distributions (i.e., histograms) of the variables to determine if they need to be transformed. You should be able to make the histograms and transform them based on what you learned during Lab 2.

par(mfrow=c(2, 4))  
hist(usAir$SO2, main="raw SO2", xlab="")  
hist(usAir$temp, main="raw temp", xlab="")  
hist(usAir$manu, main="raw manu", xlab="")  
hist(usAir$popul, main="raw popul", xlab="")  
hist(usAir$wind, main="raw wind", xlab="")  
hist(usAir$precip, main="raw precip", xlab="")  
hist(usAir$predays, main="raw predays", xlab="")

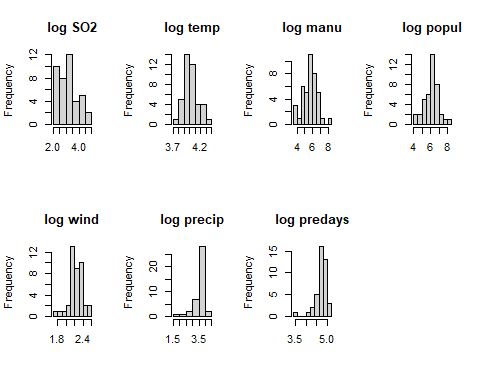


mvn(usAir, mvnTest = "mardia")

## $multivariateNormality  
## Test Statistic p value Result  
## 1 Mardia Skewness 226.612731693166 4.82491336101954e-15 NO  
## 2 Mardia Kurtosis 3.97754689564216 6.96298933924311e-05 NO  
## 3 MVN <NA> <NA> NO  
##   
## $univariateNormality  
## Test Variable Statistic p value Normality  
## 1 Anderson-Darling SO2 2.3841 <0.001 NO   
## 2 Anderson-Darling temp 0.9633 0.0136 NO   
## 3 Anderson-Darling manu 4.2925 <0.001 NO   
## 4 Anderson-Darling popul 3.4292 <0.001 NO   
## 5 Anderson-Darling wind 0.3784 0.3911 YES   
## 6 Anderson-Darling precip 0.8742 0.0228 NO   
## 7 Anderson-Darling predays 0.5175 0.1783 YES   
##   
## $Descriptives  
## n Mean Std.Dev Median Min Max 25th 75th Skew  
## SO2 41 30.048780 23.472272 26.00 8.00 110.0 13.00 35.00 1.584112608  
## temp 41 55.763415 7.227716 54.60 43.50 75.5 50.60 59.30 0.822975684  
## manu 41 463.097561 563.473948 347.00 35.00 3344.0 181.00 462.00 3.484603302  
## popul 41 608.609756 579.113023 515.00 71.00 3369.0 299.00 717.00 2.941257977  
## wind 41 9.443902 1.428644 9.30 6.00 12.7 8.70 10.60 0.002675131  
## precip 41 36.769024 11.771550 38.74 7.05 59.8 30.96 43.11 -0.692518149  
## predays 41 113.902439 26.506419 115.00 36.00 166.0 103.00 128.00 -0.550092270  
## Kurtosis  
## SO2 2.25541093  
## temp 0.09066032  
## manu 14.33200058  
## popul 10.57605759  
## wind 0.06015407  
## precip 0.49578021  
## predays 0.72033969

Wind & predays are normal - do not need transformation

logusAir = log(usAir)  
  
par(mfrow=c(2, 4))  
hist(logusAir$SO2, main="log SO2", xlab="") # \*transformation works  
hist(logusAir$temp, main="log temp", xlab="") # \*  
hist(logusAir$manu, main="log manu", xlab="") # \*  
hist(logusAir$popul, main="log popul", xlab="") # \*  
hist(logusAir$wind, main="log wind", xlab="") # does not need transformation  
hist(logusAir$precip, main="log precip", xlab="") # does not need transformation  
hist(logusAir$predays, main="log predays", xlab="") # does not need transformation

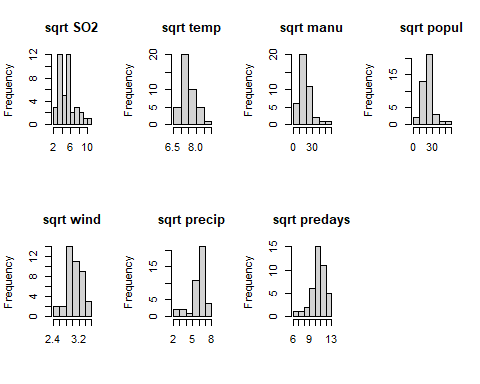


#testing normality  
mvn(logusAir, mvnTest = "mardia")

## $multivariateNormality  
## Test Statistic p value Result  
## 1 Mardia Skewness 147.330123321684 2.39626973265984e-05 NO  
## 2 Mardia Kurtosis 1.26642862677015 0.205359667379893 YES  
## 3 MVN <NA> <NA> NO  
##   
## $univariateNormality  
## Test Variable Statistic p value Normality  
## 1 Anderson-Darling SO2 0.5777 0.1247 YES   
## 2 Anderson-Darling temp 0.5964 0.114 YES   
## 3 Anderson-Darling manu 0.3850 0.3774 YES   
## 4 Anderson-Darling popul 0.5971 0.1136 YES   
## 5 Anderson-Darling wind 0.4714 0.2327 YES   
## 6 Anderson-Darling precip 3.2861 <0.001 NO   
## 7 Anderson-Darling predays 1.6096 3e-04 NO   
##   
## $Descriptives  
## n Mean Std.Dev Median Min Max 25th 75th  
## SO2 41 3.153004 0.7022985 3.258097 2.079442 4.700480 2.564949 3.555348  
## temp 41 4.013331 0.1249002 4.000034 3.772761 4.324133 3.923952 4.082609  
## manu 41 5.692094 0.9634348 5.849325 3.555348 8.114923 5.198497 6.135565  
## popul 41 6.101705 0.8044345 6.244167 4.262680 8.122371 5.700444 6.575076  
## wind 41 2.233772 0.1562515 2.230014 1.791759 2.541602 2.163323 2.360854  
## precip 41 3.525994 0.4622857 3.656873 1.953028 4.091006 3.432696 3.763755  
## predays 41 4.701922 0.2824149 4.744932 3.583519 5.111988 4.634729 4.852030  
## Skew Kurtosis  
## SO2 0.3148634 -0.9176356  
## temp 0.5505350 -0.2631236  
## manu -0.1188929 0.0250294  
## popul -0.1607179 0.1726590  
## wind -0.5179759 0.5599034  
## precip -1.9614768 3.6965190  
## predays -1.7438885 4.3363444

Everything normal except precip & predays (does not need transformation anyway), and log not appropriate for popul and manu (count data). Log for precip improves normality than if it were to be left untransformed

sqrtusAir = sqrt(usAir)  
  
par(mfrow=c(2, 4))  
hist(sqrtusAir$SO2, main="sqrt SO2", xlab="") #log better  
hist(sqrtusAir$temp, main="sqrt temp", xlab="") #log better  
hist(sqrtusAir$manu, main="sqrt manu", xlab="") #log better but not appropriate for count  
hist(sqrtusAir$popul, main="sqrt popul", xlab="") #log better but not appropriate for count  
hist(sqrtusAir$wind, main="sqrt wind", xlab="") # does not need transformation - period  
hist(sqrtusAir$precip, main="sqrt precip", xlab="") # does not need transformation - not that much diff from raw & log is worse  
hist(sqrtusAir$predays, main="sqrt predays", xlab="") # does not need transformation



#testing normality  
mvn(sqrtusAir, mvnTest = "mardia")

## $multivariateNormality  
## Test Statistic p value Result  
## 1 Mardia Skewness 150.793018272587 1.07207656634339e-05 NO  
## 2 Mardia Kurtosis 1.00256610035013 0.316070258885355 YES  
## 3 MVN <NA> <NA> NO  
##   
## $univariateNormality  
## Test Variable Statistic p value Normality  
## 1 Anderson-Darling SO2 1.1362 0.0050 NO   
## 2 Anderson-Darling temp 0.7655 0.0428 NO   
## 3 Anderson-Darling manu 1.1010 0.0061 NO   
## 4 Anderson-Darling popul 1.0187 0.0099 NO   
## 5 Anderson-Darling wind 0.3903 0.3665 YES   
## 6 Anderson-Darling precip 1.8076 0.0001 NO   
## 7 Anderson-Darling predays 0.9184 0.0177 NO   
##   
## $Descriptives  
## n Mean Std.Dev Median Min Max 25th 75th  
## SO2 41 5.146869 1.9098376 5.099020 2.828427 10.488088 3.605551 5.916080  
## temp 41 7.452804 0.4739282 7.389181 6.595453 8.689074 7.113368 7.700649  
## manu 41 19.252281 9.7343927 18.627936 5.916080 57.827329 13.453624 21.494185  
## popul 41 22.839065 9.4425391 22.693611 8.426150 58.043087 17.291616 26.776856  
## wind 41 3.064312 0.2350320 3.049590 2.449490 3.563706 2.949576 3.255764  
## precip 41 5.962347 1.1180007 6.224147 2.655184 7.733046 5.564171 6.565821  
## predays 41 10.590310 1.3384570 10.723805 6.000000 12.884099 10.148892 11.313708  
## Skew Kurtosis  
## SO2 0.9134739 0.1646949  
## temp 0.6876931 -0.1088396  
## manu 1.6434266 4.1836591  
## popul 1.3595340 3.1252514  
## wind -0.2508710 0.2306118  
## precip -1.3222345 1.6989584  
## predays -1.0987299 1.9939477

Sqrt improves the normality of manu and popul and appropriate transformation for the data class. Precip seems to be more normal raw than transformed.

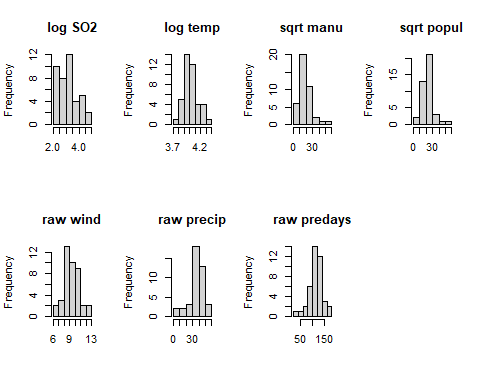
transusAir = usAir %>%  
 mutate(logSO2 = log(SO2), logtemp = log(temp), sqrtmanu = sqrt(manu), sqrtpop = sqrt(popul)) %>%  
 select(-c("SO2", "temp", "manu", "popul"))  
  
head(transusAir, 4)

## wind precip predays logSO2 logtemp sqrtmanu sqrtpop  
## Albany 8.8 33.36 135 3.828641 3.862833 6.63325 10.77033  
## Albuquerque 8.9 7.77 58 2.397895 4.039536 6.78233 15.62050  
## Atlanta 9.1 48.34 115 3.178054 4.119037 19.18333 22.29350  
## Baltimore 9.6 41.31 111 3.850148 4.007333 25.00000 30.08322

#testing normality  
mvn(transusAir, mvnTest = "mardia")

## $multivariateNormality  
## Test Statistic p value Result  
## 1 Mardia Skewness 138.095747086503 0.000184359228625694 NO  
## 2 Mardia Kurtosis 0.448796824374274 0.653578229775731 YES  
## 3 MVN <NA> <NA> NO  
##   
## $univariateNormality  
## Test Variable Statistic p value Normality  
## 1 Anderson-Darling wind 0.3784 0.3911 YES   
## 2 Anderson-Darling precip 0.8742 0.0228 NO   
## 3 Anderson-Darling predays 0.5175 0.1783 YES   
## 4 Anderson-Darling logSO2 0.5777 0.1247 YES   
## 5 Anderson-Darling logtemp 0.5964 0.1140 YES   
## 6 Anderson-Darling sqrtmanu 1.1010 0.0061 NO   
## 7 Anderson-Darling sqrtpop 1.0187 0.0099 NO   
##   
## $Descriptives  
## n Mean Std.Dev Median Min Max 25th  
## wind 41 9.443902 1.4286442 9.300000 6.000000 12.700000 8.700000  
## precip 41 36.769024 11.7715498 38.740000 7.050000 59.800000 30.960000  
## predays 41 113.902439 26.5064189 115.000000 36.000000 166.000000 103.000000  
## logSO2 41 3.153004 0.7022985 3.258097 2.079442 4.700480 2.564949  
## logtemp 41 4.013331 0.1249002 4.000034 3.772761 4.324133 3.923952  
## sqrtmanu 41 19.252281 9.7343927 18.627936 5.916080 57.827329 13.453624  
## sqrtpop 41 22.839065 9.4425391 22.693611 8.426150 58.043087 17.291616  
## 75th Skew Kurtosis  
## wind 10.600000 0.002675131 0.06015407  
## precip 43.110000 -0.692518149 0.49578021  
## predays 128.000000 -0.550092270 0.72033969  
## logSO2 3.555348 0.314863354 -0.91763558  
## logtemp 4.082609 0.550534971 -0.26312362  
## sqrtmanu 21.494185 1.643426644 4.18365909  
## sqrtpop 26.776856 1.359533999 3.12525141

par(mfrow=c(2,4))  
  
hist(transusAir$logSO2, main="log SO2", xlab="")   
hist(transusAir$logtemp, main="log temp", xlab="")   
hist(transusAir$sqrtmanu, main="sqrt manu", xlab="")   
hist(transusAir$sqrtpop, main="sqrt popul", xlab="")   
hist(transusAir$wind, main="raw wind", xlab="")  
hist(transusAir$precip, main="raw precip", xlab="")  
hist(transusAir$predays, main="raw predays", xlab="")



## **What variable did you transform, and what transformation did you use? (15 pts)**

I log transformed SO2 and temp, and square root transformed manu and popul. The rest were left as is, log and square root transformations exaggerated skewness.

If you do transform any variables, use the transformed data matrix going forward.

Next, apply a z-score standardization:

ZusAir<-scale(transusAir) #especially important to scale bc in very different units now

# Running the PCA:

You are going to use the package princomp function in the stats package. Take some time to read about this function:

?princomp

Run the princomp function:

usAir\_pca <- princomp(ZusAir, cor = F)

cor = F, because you are using the *covariance* matrix instead of the *correlation* matrix.

**It should be noted that the covariance matrix of a z-standardized data matrix is equivalent to the correlation matrix of the unscaled data.**

Let’s look at a summary of our PCA:

summary(usAir\_pca)

## Importance of components:  
## Comp.1 Comp.2 Comp.3 Comp.4 Comp.5  
## Standard deviation 1.5814860 1.2434605 1.1749147 0.9831819 0.54899114  
## Proportion of Variance 0.3662322 0.2264070 0.2021336 0.1415447 0.04413229  
## Cumulative Proportion 0.3662322 0.5926392 0.7947728 0.9363175 0.98044976  
## Comp.6 Comp.7  
## Standard deviation 0.28863711 0.224059003  
## Proportion of Variance 0.01219917 0.007351071  
## Cumulative Proportion 0.99264893 1.000000000

Notice that the summary gives the standard deviation instead of the eigenvalue (variance). Let’s calculate the eigenvalues using what we know about the relationship between standard deviation and variance (var = sd^2):

eigenVal<- (usAir\_pca$sdev\*sqrt(41/40))^2

The *sqrt(41/40)* is to correct for the fact that princomp calculates variances with the divisor N instead of N-1 as is customary. This adjustment will allow direct comparison with “hand” calculated eigenvalues using the function eigen below. Note that these hand calculations are just to help you understand what is going on ‘under the hood’ in the model.

Let’s make the PCA table with the eigenvalues instead of the standard deviations:

propVar<-eigenVal/sum(eigenVal) #recalculating column to include eigenvalue  
cumVar<-cumsum(propVar) #recalculating column to include eigenvalue  
pca\_Table<-t(rbind(eigenVal,propVar,cumVar))  
pca\_Table

## eigenVal propVar cumVar  
## Comp.1 2.56362542 0.366232202 0.3662322  
## Comp.2 1.58484892 0.226406989 0.5926392  
## Comp.3 1.41493506 0.202133581 0.7947728  
## Comp.4 0.99081287 0.141544696 0.9363175  
## Comp.5 0.30892606 0.044132294 0.9804498  
## Comp.6 0.08539417 0.012199167 0.9926489  
## Comp.7 0.05145750 0.007351071 1.0000000

**This calculation and table are just to show you that the eigenvalues and the output from princomp, the stadard deviations, are the same thing. YOU DO NOT NEED TO DO THIS WHEN YOU RUN A PCA - it is just to help you gain a deeper understanding of the model.**

the factor loadings from princomp:

loadings(usAir\_pca)

##   
## Loadings:  
## Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7  
## wind 0.285 0.143 0.844 0.368 0.208   
## precip -0.404 -0.688 0.185 0.215 -0.515 -0.136  
## predays 0.342 -0.591 -0.151 -0.534 0.454 0.108  
## logSO2 0.450 -0.258 -0.482 0.651 0.237 -0.112  
## logtemp -0.387 0.234 -0.583 0.211 0.614 0.183  
## sqrtmanu 0.518 0.358 -0.223 -0.216 0.703  
## sqrtpop 0.426 0.481 -0.297 -0.247 -0.655  
##   
## Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7  
## SS loadings 1.000 1.000 1.000 1.000 1.000 1.000 1.000  
## Proportion Var 0.143 0.143 0.143 0.143 0.143 0.143 0.143  
## Cumulative Var 0.143 0.286 0.429 0.571 0.714 0.857 1.000

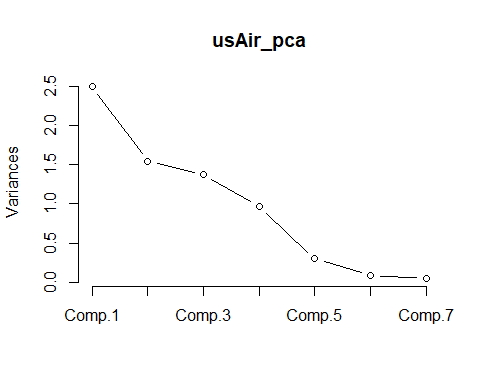
and the factor scores:

scores(usAir\_pca)

## Determining number of axes to keep

You now have 7 PC axes. Which ones give us vital information and which ones can we toss? One method for selecting the number of Axes is a **Scree plot**:

plot(usAir\_pca, type="lines")

 \*\*\_\*\* Based on the scree plot, it seems like we can keep the first five since they explain the majoritiy of the variation and the variation seems to level off around component 5.

How about the **latent root criterion (i.e., keep components with eigenvalues > 1)** and the **relative percent variance criteria**. Check the summary of the PCA explore this:

summary(usAir\_pca)

## Importance of components:  
## Comp.1 Comp.2 Comp.3 Comp.4 Comp.5  
## Standard deviation 1.5814860 1.2434605 1.1749147 0.9831819 0.54899114  
## Proportion of Variance 0.3662322 0.2264070 0.2021336 0.1415447 0.04413229  
## Cumulative Proportion 0.3662322 0.5926392 0.7947728 0.9363175 0.98044976  
## Comp.6 Comp.7  
## Standard deviation 0.28863711 0.224059003  
## Proportion of Variance 0.01219917 0.007351071  
## Cumulative Proportion 0.99264893 1.000000000

## **Question 2: How many axes you should keep and why? (20 points)**

Based on the latent root criterion, we should keep the first three components since after that eigenvalues are < 1.

## Significance of factor loadings.

While many use the “rule of thumb” where a loading > 0.30 dictates an “important” variable. Another method for determining significance of factor loadings is bootstrapping. Details and comparisons of the many way to assess significance of factor loadings are presented in Peres-Neto et al. (2003), which is supplemental reading for this week. Here we will run the method that they found to have the lowest type I error rates, Bootstrapped eigenvector. For reference, this is the method 6 in Peres-Neto et al. (2003).

sigpca2<-function (x, permutations=1000, ...)  
{  
 pcnull <- princomp(x, ...)  
 res <- pcnull$loadings  
 out <- matrix(0, nrow=nrow(res), ncol=ncol(res))  
 N <- nrow(x)  
 for (i in 1:permutations) { #resample from x, princomp() it and compare PC info to orig PC, do this 1k times  
 pc <- princomp(x[sample(N, replace=TRUE), ], ...) #randomly sample (w/ replacment) from the observed data pool (x), with the same number of rows (N) and no change to the columns  
 pred <- predict(pc, newdata = x)  
 r <- cor(pcnull$scores, pred)  
 k <- apply(abs(r), 2, which.max)  
 reve <- sign(diag(r[k,]))  
 sol <- pc$loadings[ ,k]  
 sol <- sweep(sol, 2, reve, "\*")  
 out <- out + ifelse(res > 0, sol <= 0, sol >= 0)  
 }  
 out/permutations  
}  
  
set.seed(4) #so we get the same random sample   
   
sigpca2(ZusAir, permutations=1000) # Use function with the scaled data as input

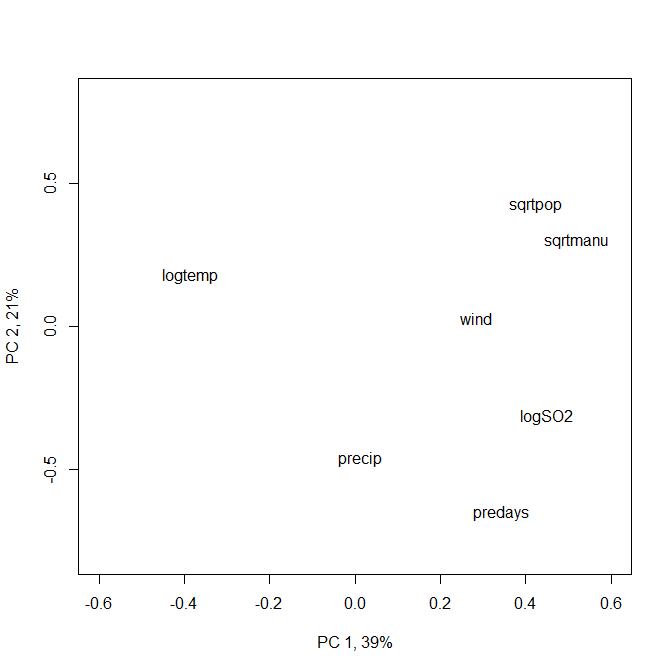
## Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7  
## wind 0.035 0.328 0.252 0.238 0.034 0.031 0.427  
## precip 0.481 0.337 0.263 0.247 0.037 0.024 0.193  
## predays 0.042 0.222 0.420 0.404 0.009 0.023 0.227  
## logSO2 0.024 0.247 0.473 0.199 0.009 0.042 0.187  
## logtemp 0.026 0.339 0.247 0.467 0.025 0.022 0.191  
## sqrtmanu 0.029 0.203 0.322 0.314 0.299 0.191 0.054  
## sqrtpop 0.081 0.203 0.283 0.432 0.019 0.353 0.047

# THE FOLLOWING IS INCLUDED IN THE FUNCTION BUT FOR ONE ITERATION  
#Piece-by-piece (read along step-by-step with pg. 2350 section 6) Bootstrapped eigenvector (V vectors)” of Peres-Neto et al. 2003.  
pcnull<-princomp(ZusAir) #  
res <- pcnull$loadings  
out <- matrix(0, nrow=nrow(res), ncol=ncol(res))  
N <- nrow(ZusAir)  
pc<-princomp(ZusAir[sample(N, replace=TRUE), ])  
pred <- predict(pc, newdata = ZusAir)   
r <- cor(pcnull$scores, pred)  
k <- apply(abs(r), 2, which.max)  
reve <- sign(diag(r[k,]))  
sol <- pc$loadings[ ,k]  
sol <- sweep(sol, 2, reve, "\*")  
out <- out + ifelse(res > 0, sol <= 0, sol >= 0)

## PCA plots

Plot out the factor loadings for the first 2 PC axes:

plot(usAir\_pca$loadings, type="n", xlab="PC 1, 39%", ylab="PC 2, 21%", ylim=c(-.8,.8), xlim=c(-.6,.6))  
text(usAir\_pca$loadings, labels=as.character(colnames(ZusAir)), pos=1, cex=1)



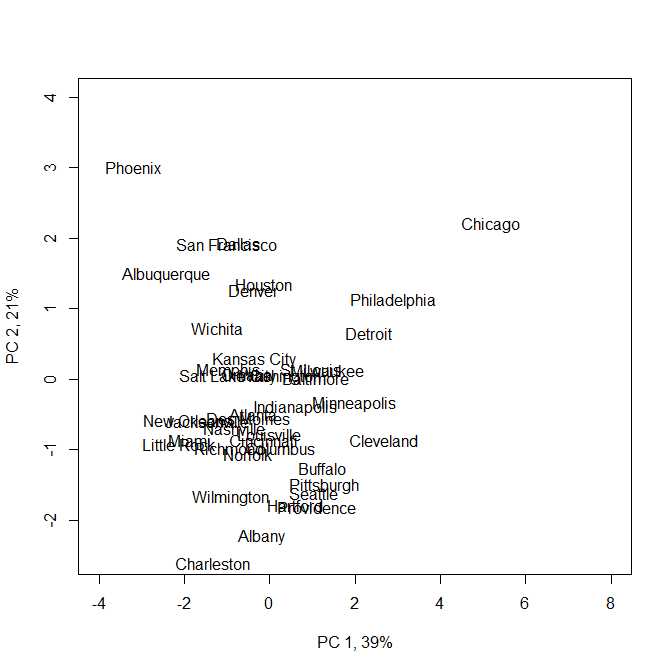
## **Question 3: How do you interpret these axes? Come up with a name for each. (20 pts)**

The axes indicate how strongly each characteristic (variable vector) influence the variation (principal components). PC 1 could be deemed the “anthropogenic” axis where the majority of the variation is explained by the population (sqrtpopul), manufacturing enterprises (sqrtmanu) and the air content (logSO2), which are all direct anthropogenic activities. PC 1 can be deemed the “temperature” axis since only temperature (logtemp) is really dominating that axis.

**Close the plot window after viewing**

Let’s now plot the PC score for each sample (city):

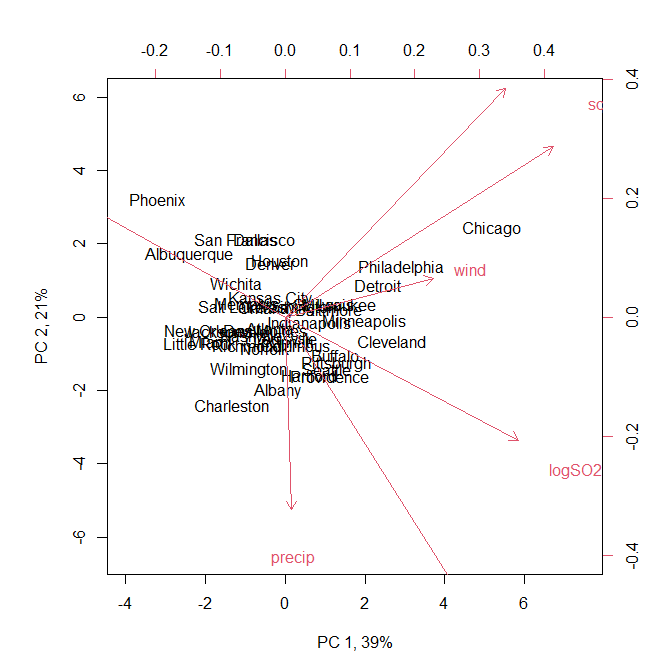
plot(usAir\_pca$scores,type="n",xlab="PC 1, 39%", ylab="PC 2, 21%",ylim=c(-2.5,4), xlim=c(-4,8))  
text(usAir\_pca$scores, labels=as.character(rownames(ZusAir)), pos=1, cex=1)



And now all together in a biplot:

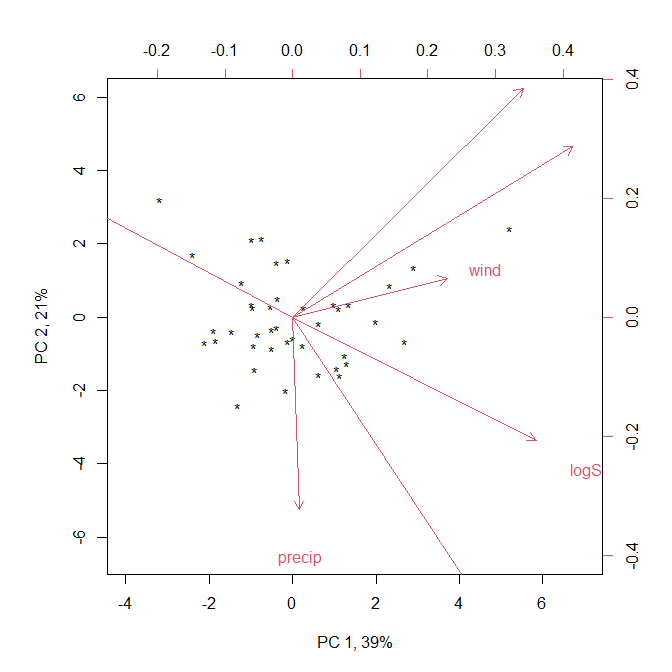
?biplot

biplot(usAir\_pca$scores, usAir\_pca$loading, xlab="PC 1, 39%", ylab="PC 2, 21%", expand= 4, ylim=c(-6.5,6), xlim=c(-4,7.5))



to replace city names with a symbol:

biplot(usAir\_pca$scores, usAir\_pca$loading, expand= 4, xlabs= rep("\*",41), xlab="PC 1, 39%", ylab="PC 2, 21%", ylim=c(-6.5,6), xlim=c(-4,7))



# Eigen Analysis

You can also just simply use the Eigen analysis function, eigen and calculate your own scores by hand. Note that I am showing this for illustration for those of you who want to have a deeper understanding of the method.

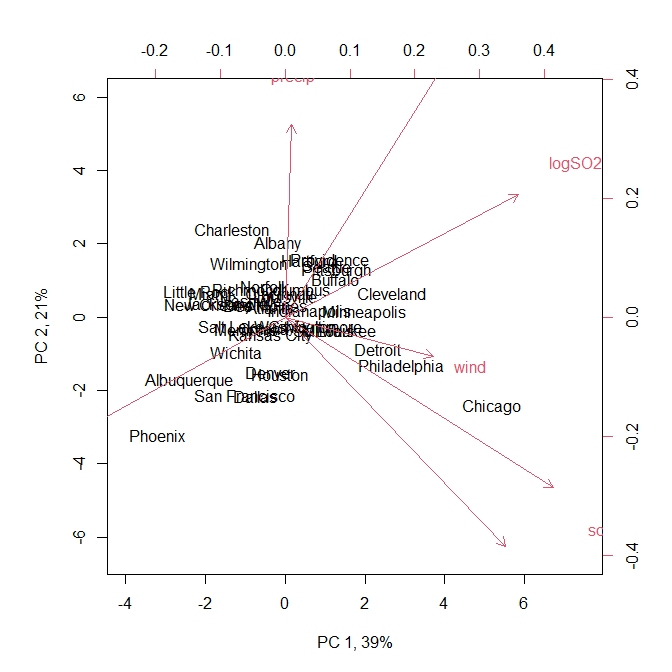
eig<-eigen(cov(ZusAir))

Extract the first two eigenvectors (because that is what we are interested in plotting):

eigVec<-as.matrix(eig$vectors[,1:2])  
rownames(eigVec) <- rownames(cov(ZusAir))

Then simply multiply each eigenvector times the matrix of standardized observation values (ZusAir) and plot!

scores<-t(rbind(eigVec[,1]%\*%t(ZusAir),eigVec[,2]%\*%t(ZusAir)))#hand calculated scores  
  
# and plot  
  
biplot(scores,eigVec,xlab="PC 1, 39%", ylab="PC 2, 21%",expand= 4, ylim=c(-6.5,6), xlim=c(-4,7.5))



# My Data set

## **Question 4: Does your individual dataset (the one used in Lab 2) meet the assumptions of PCA? Is PCA an analysis you could use on your data? (40 pts)**

My data set does not meet the assumptions of multivariate normality (even after transformations), and so a PCA is not appropriate for this data set.