Data Science Methods - Assignment 1

M. Alberti, 2020162 N.R. Ceschin

February 21, 2020

First we upload all relevant libraries:

```
library(readxl)
library(ggplot2)
library(ggfortify)
library(dplyr)
library(tidyr)
library(RCurl)
library(ggrepel)
```

Upload dataset:

```
setwd("C:/Users/Mr Nobody/Desktop/Uni/EME/Data science Methods/Assignments")
#setwd("~/Tilburg/Courses/Data Science Methods/Assignment1/DATA-SCIENCE-ASSIGNMENTS")
data<-read_excel("env_air_emis.xls")
x <- getURL("https://raw.githubusercontent.com/AlbertiMarco/DATA-SCIENCE-ASSIGNMENTS/master/EU%20labels
EU_labels<- read.csv(text = x, header = FALSE ,sep=";") #import country tags to make plots more readabl
rownames(EU_labels)<-EU_labels[[1]]
```

After a quick glimpse of the data we realized that information for the five pollutant are presented in separated consecutive tables, the separation contains some information in the first column and NA cells in the rest. To be sure not to drop NAs in the middle of the dataset, we first proceed to drop all raws containing at least 5 NA values and we assign to *df*:

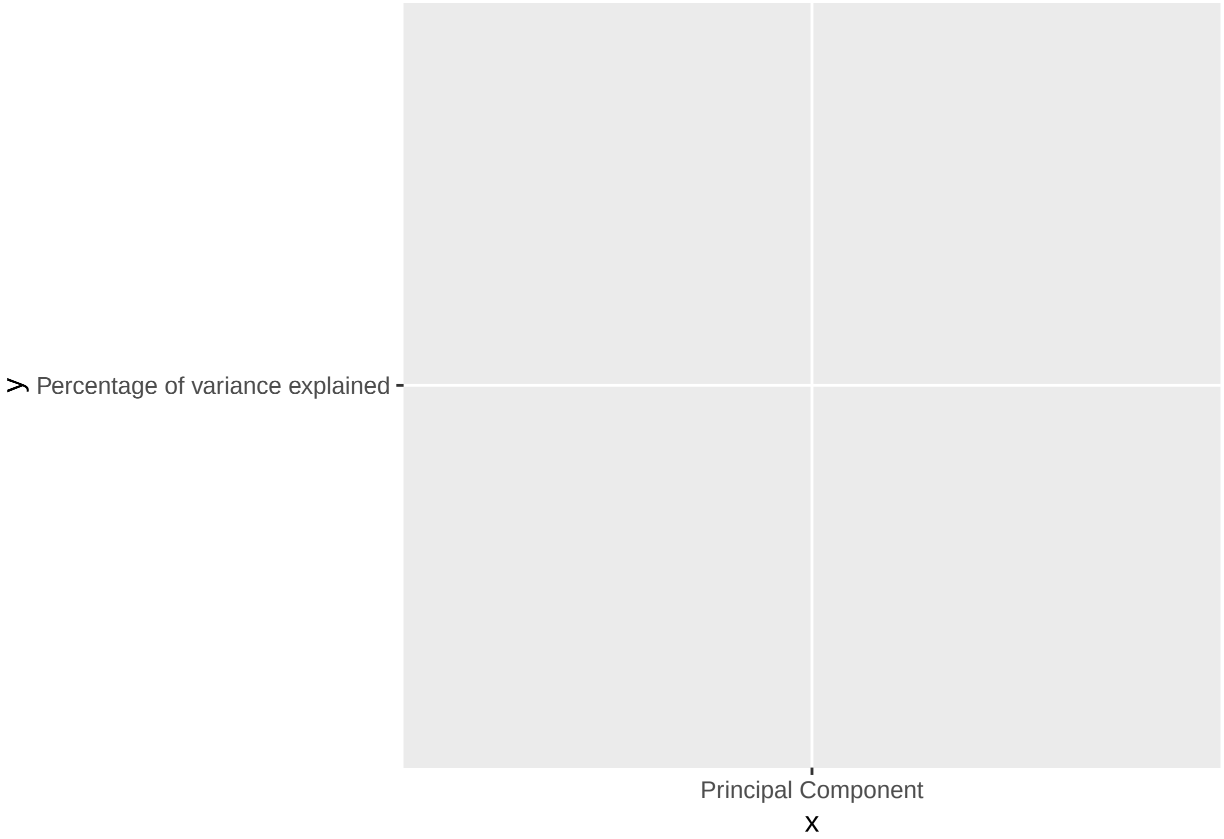
```
dim(data)
df<-data[rowSums(is.na(data))<length(data)-5,]
#df<-data[complete.cases(data), ]
```

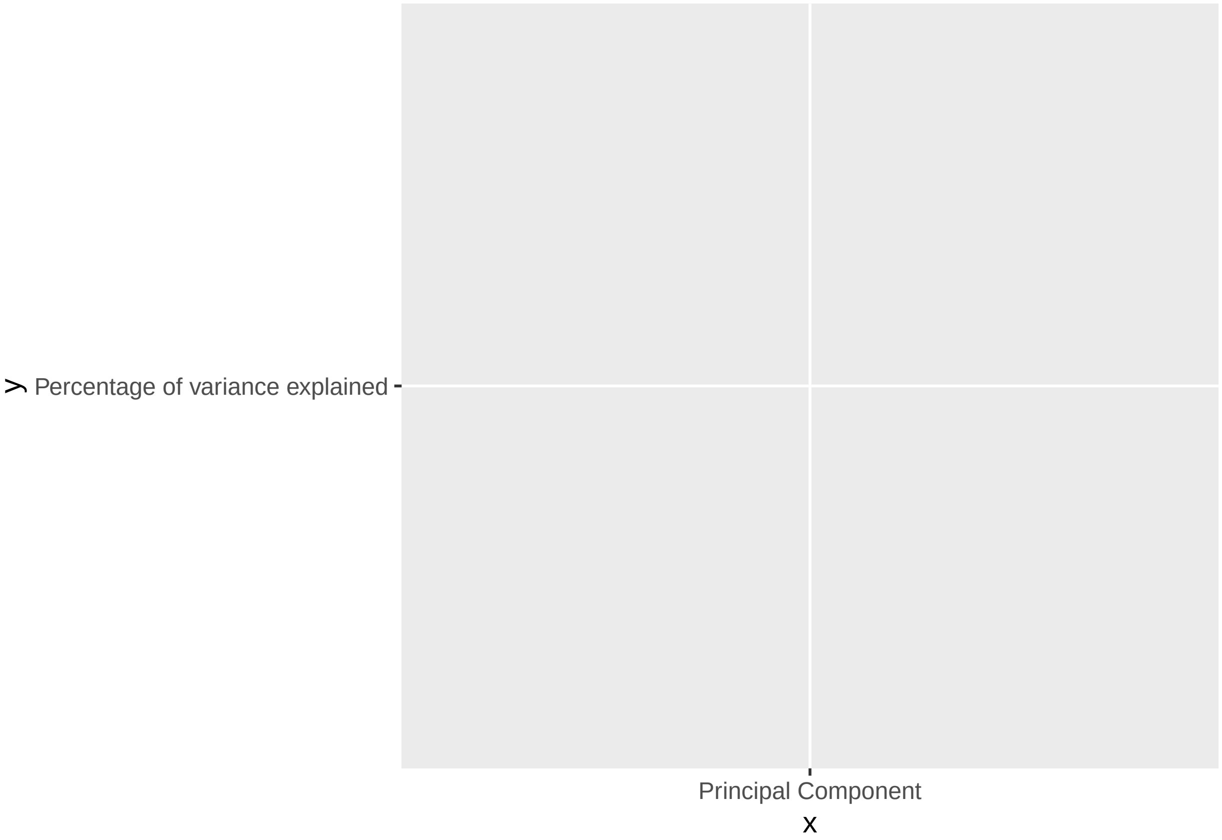
Given the data structure and the subpoints a-c requests, we decided that the optimal approach would be looping over the chunks of data containing information for each pollutant, producing without repeting the code the outputs all in one step. First we create some variables that will be used in the loop:

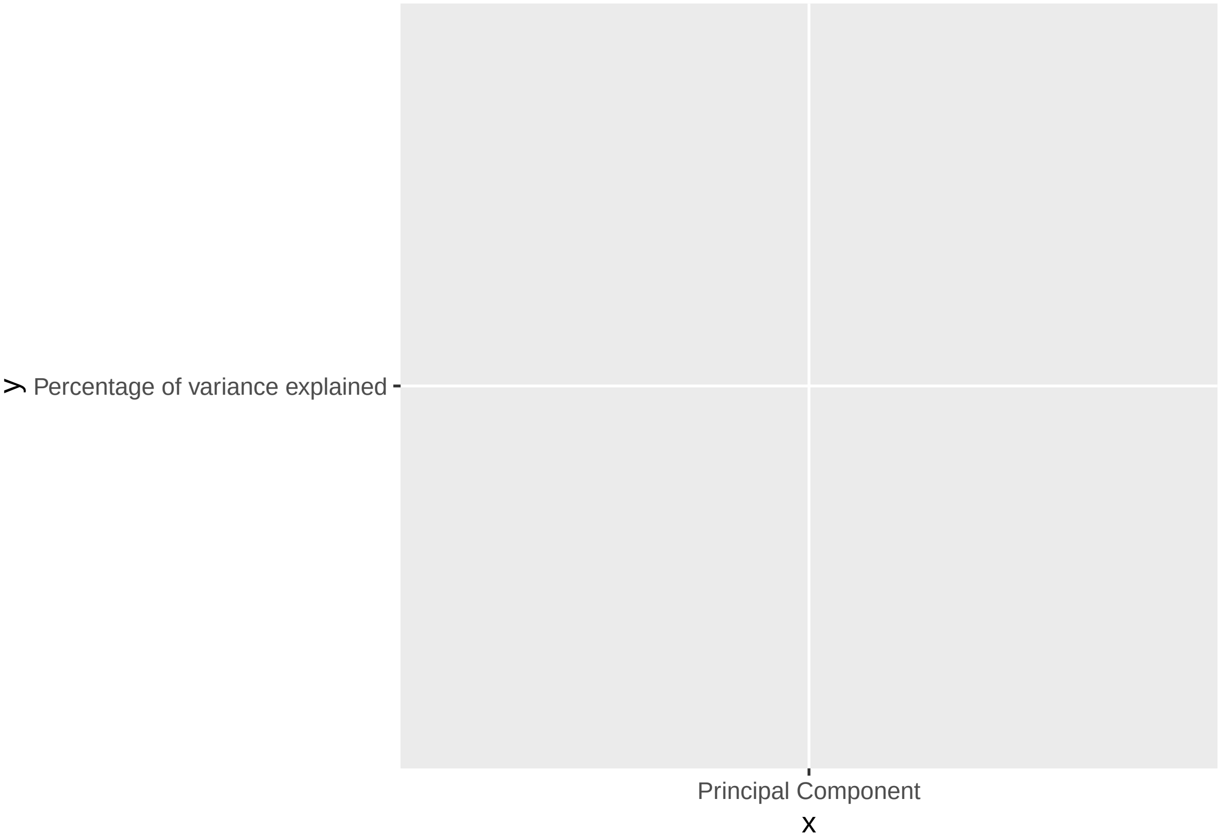
```
#build 'index' for your loop
interval<-c(1,30,59,88,117)   #number of the first row of each individidual dataset
pollutants<-c("ammonia","nmvoc","smallpart","largepart","sulphur")
index<-data.frame(interval,pollutants)

PC1<-data.frame(matrix(ncol=5,nrow=28))  #empty data frames that will be filled with the scores
PC2<-data.frame(matrix(ncol=5,nrow=28))  # of the PC 1 and 2 for each pollutant and country
```

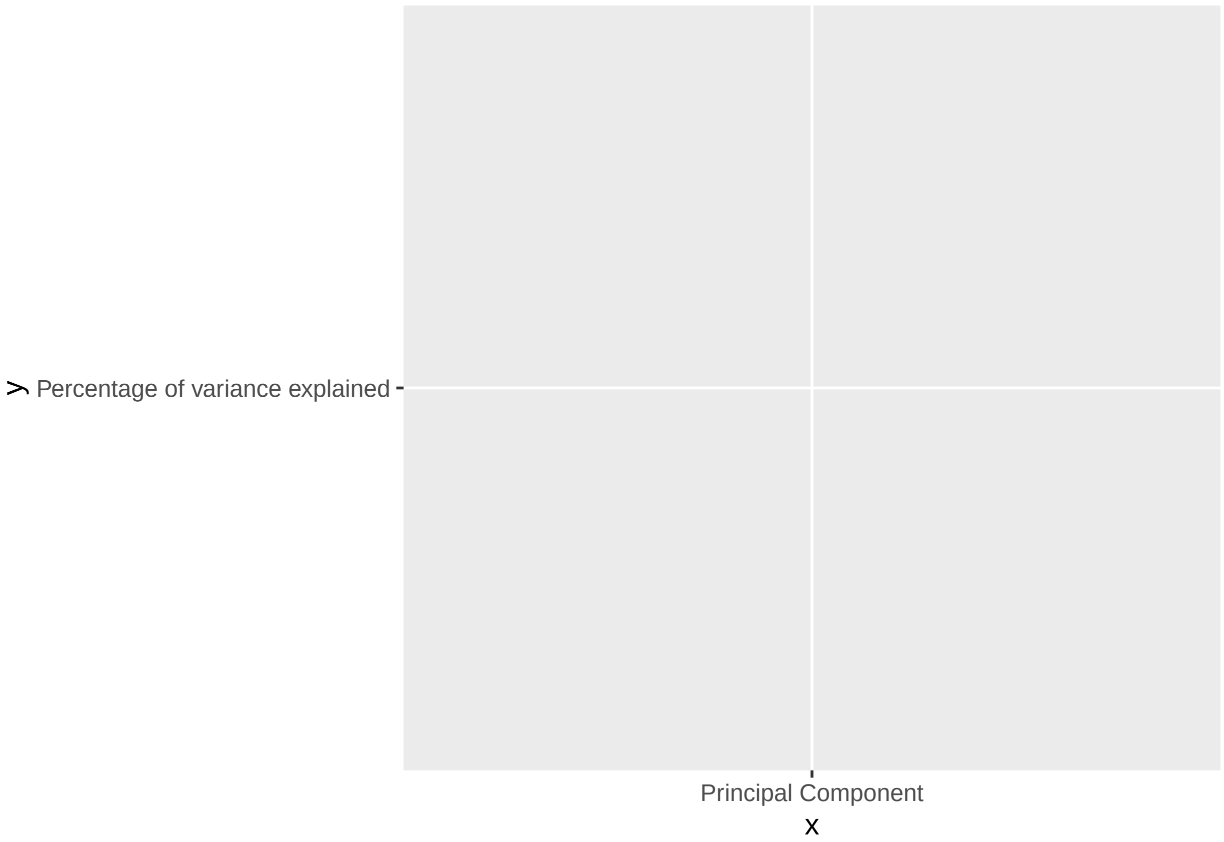
Then we run the loop that: 1) Selects the chunk of the dataframe corresponding to one pollutant and puts it into shape 2) Runs PCA on the reduced dataframe and produces the plots of the first two PCs 3) Produces a scree plot
4) Computes the Bayesian Informatio Criteria and prints the optimal number of PCs according to this method
5) Stores plots and loadings for future use

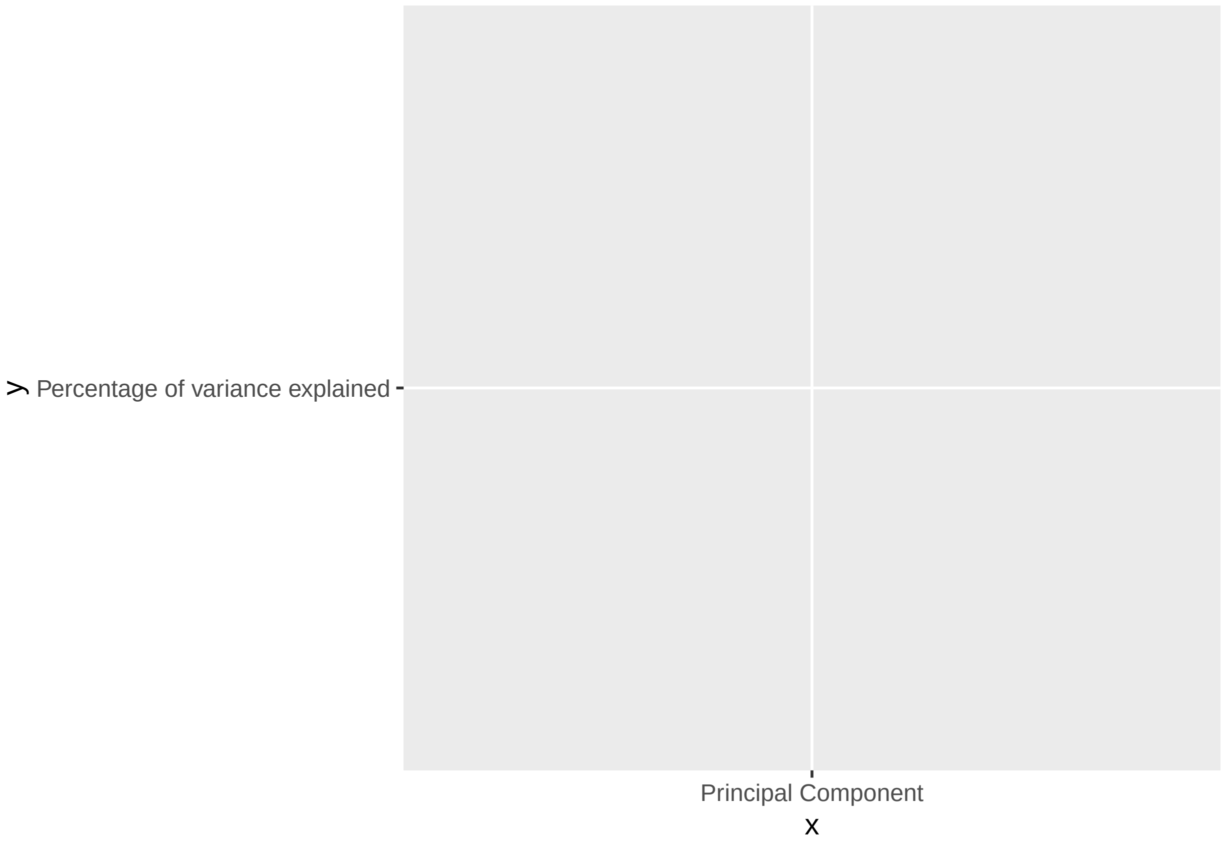
```
mytheme <- theme(plot.title= element_text(face="bold",colour = "antiquewhite4",size = (16),hjust = 0.5)

for (i in 1:5){
  #data chunk preparation
  begin<-index[i,1]
  end<-index[i,1]+28      #each chunk has 27 countries plus the first raw with years
  dfx<-df[begin:end,] #slice portion of the dataframe, 'according to begin' and 'end'
  dfx<-as.data.frame(dfx) #rename first column with the name of the pollutant
  colnames(dfx)<-dfx[1,]         #set first column as observations' names and first row as       variab
  rownames(dfx)<-dfx[,1]
  dfx<-dfx[c(2:29),c(2:29)]      #drop first column and obtain the final datset
  if (sum(mapply(grepl,rownames(EU_labels),rownames(dfx)))==length(dfx)) {
  rownames(dfx)<- EU_labels[[2]]
    } #Substitute name with short labels of the appropriate country
  dfx<-as.data.frame(t(dfx))     #convert factor columns into numeric to apply prcomp
  indx <- sapply(dfx, is.factor)
  dfx[indx] <- lapply(dfx[indx], function(x) as.numeric(as.character(x)))


  #Principal Component Analysis
  pr.out<-prcomp(dfx, scale=TRUE)
  #print(pr.out$rotation[,1:2])           # print first two PC loadings and plot first two PC
  graph<-autoplot(pr.out,variance_percentage=FALSE,loadings=TRUE,
           loadings.label=TRUE,loadings.label.repel=TRUE,loadings.colour="coral",loadings.label.size=3,

  pve= 100* (pr.out$sdev ^2)/ sum(pr.out$sdev ^2)  #screeplot
  scree<- ggplot(data.frame(pve),aes("Principal Component", "Percentage of variance explained"))
  print(scree)
    #plot(pve , type ="o", ylab="PVE ", xlab=" Principal Component ",
       #col =" blue",axes = F)
  #axis(side = 1, at = seq(from=0,to=30))
  #axis(side = 2, at = seq(from=0,to=100,by=5))
  #title(paste("Scree plot for PCs of",toString(index[i,2]),"pollutant"))
  #grid()

  #compute vector of BIC for first 27 principal components
  BIC<-c(1:27)   #initialize a numeric vector to be filled with BIC(k) values. set max k=p-1
  for (j in 1:27) {
    f<-t(pr.out$rotation[,1:j])%*%pr.out$x[,1:j] #compute aF in X=aF+e
    res_mat<-dfx[,1:j]-f                  #compute matrix of residuals
    res_mat_sq<-res_mat*res_mat                  #square residuals
    if (j==1){
      res<-(sum(res_mat_sq)/(dim(dfx)[1]*dim(dfx)[2]))
    } else{
    res<-(sum(rowSums(res_mat_sq))/(dim(dfx)[1]*dim(dfx)[2]))         #residuals sum of squares
    }
    k<-j
    BICk<-log(res)+k*(log(28^2))/(28^2) #BIC for each k
    BIC[j]<-BICk                                 #fill BIC vector at each iteration
    }
  min<-min(BIC)
  num_pc<-match(min,BIC)                         #find and print k, the index of the min of BIC
  cat("According to the BIC criterion, the optimal number of principal components is ", num_pc)
  ###potential issue: smallest value for BIC is always the one with ###
  ###the max number of principal components...strange!I checked the calculations###
  ###and they seem fine. I think the issue is that the penalty part of BIC is really###
  ###trivial compared to the log(SSR) part####

  #save first two PC in separate dataset for point d)
  PC1[i]<-pr.out$x[,1]
  colnames(PC1)[i]<-as.character(index[i,2])
  PC2[i]<-pr.out$x[,2]
  colnames(PC2)[i]<-as.character(index[i,2])

  #save relevant objects with their respective name
  assign(paste0("BIC_", index[i,2]), BIC)
  assign(paste0("df_", index[i,2]), dfx)
  assign(paste0("prcomp_",index[i,2]),pr.out)
  assign(paste0("Screeplot_",index[i,2]),scree)
  assign(paste0("PC1-PC2_",index[i,2]),graph)
  #remove non relevant objects
  rm(dfx)
  rm(BIC)
  rm(pr.out)
}
```




y
Percentage of variance explained
Principal Component
x


y
Percentage of variance explained
Principal Component
x


y
Percentage of variance explained
Principal Component
x



Now to comment on the results we can called the corresponding stored object.

d)

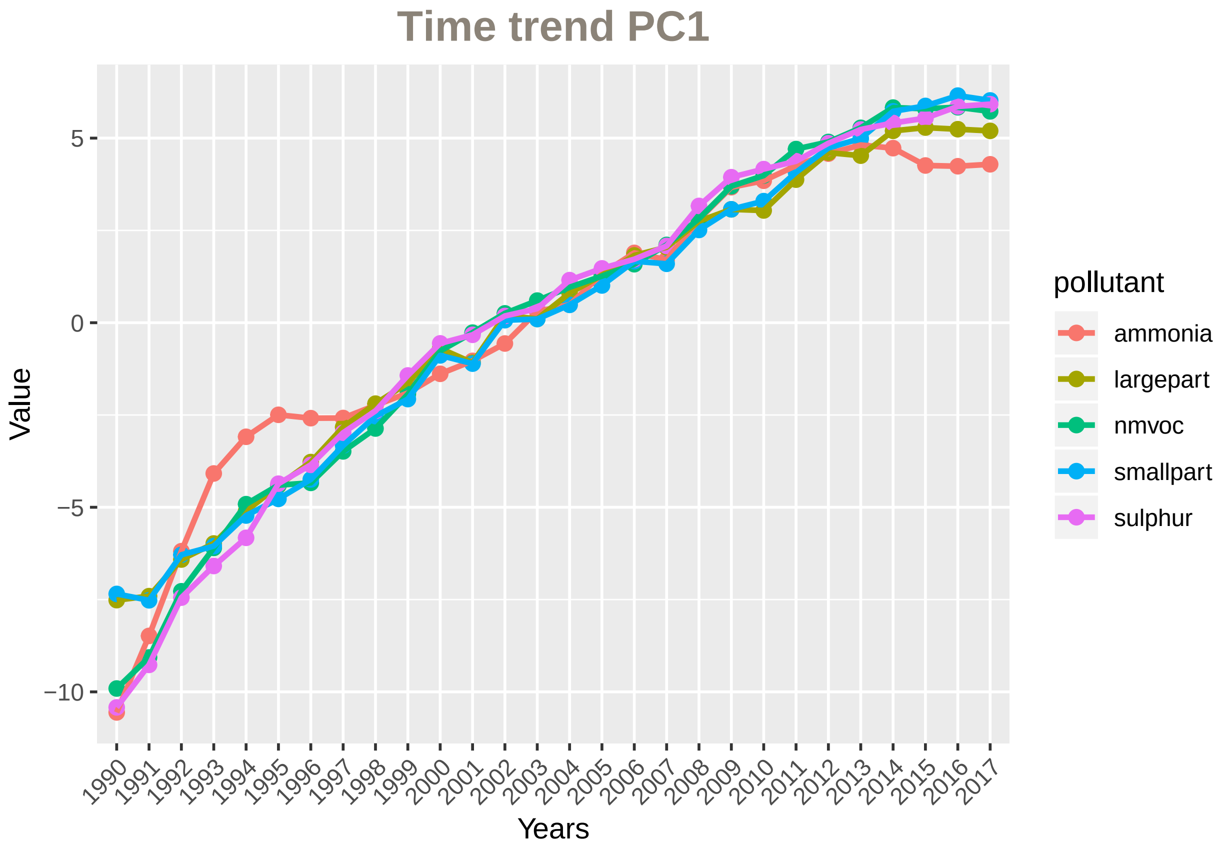
We plot the first 2 principal components for all the five pollutants over the time interval.

Code to produce the plots:

```
theme2<-theme(axis.text.x = element_text(angle = 45, hjust=1))
PC1["years"]<-c(1990:2017)
PC1<-gather(PC1, `ammonia`, `largepart`, `nmvoc`,`smallpart`, `sulphur`, key = "pollutant", value = "va
PC1_plot<-ggplot(PC1,aes(x=factor(years),y=value, group=pollutant,color=pollutant))+
  geom_point(size = 2.25) +  geom_line(size = 1) +theme2 +mytheme+
  labs(title = "Time trend PC1",x="Years",y="Value")

PC2["years"]<-c(1990:2017)
PC2<-gather(PC2, `ammonia`, `largepart`, `nmvoc`,      #transform data to be plotted
            `smallpart`, `sulphur`, key = "pollutant", value = "value")
PC2_plot<-ggplot(PC2,aes(x=factor(years),y=value, group=pollutant,color=pollutant))+
  geom_point(size = 2.25) +  geom_line(size = 1) +theme2 +mytheme+
  labs(title = "Time trend PC2",x="Years",y="Value")
```

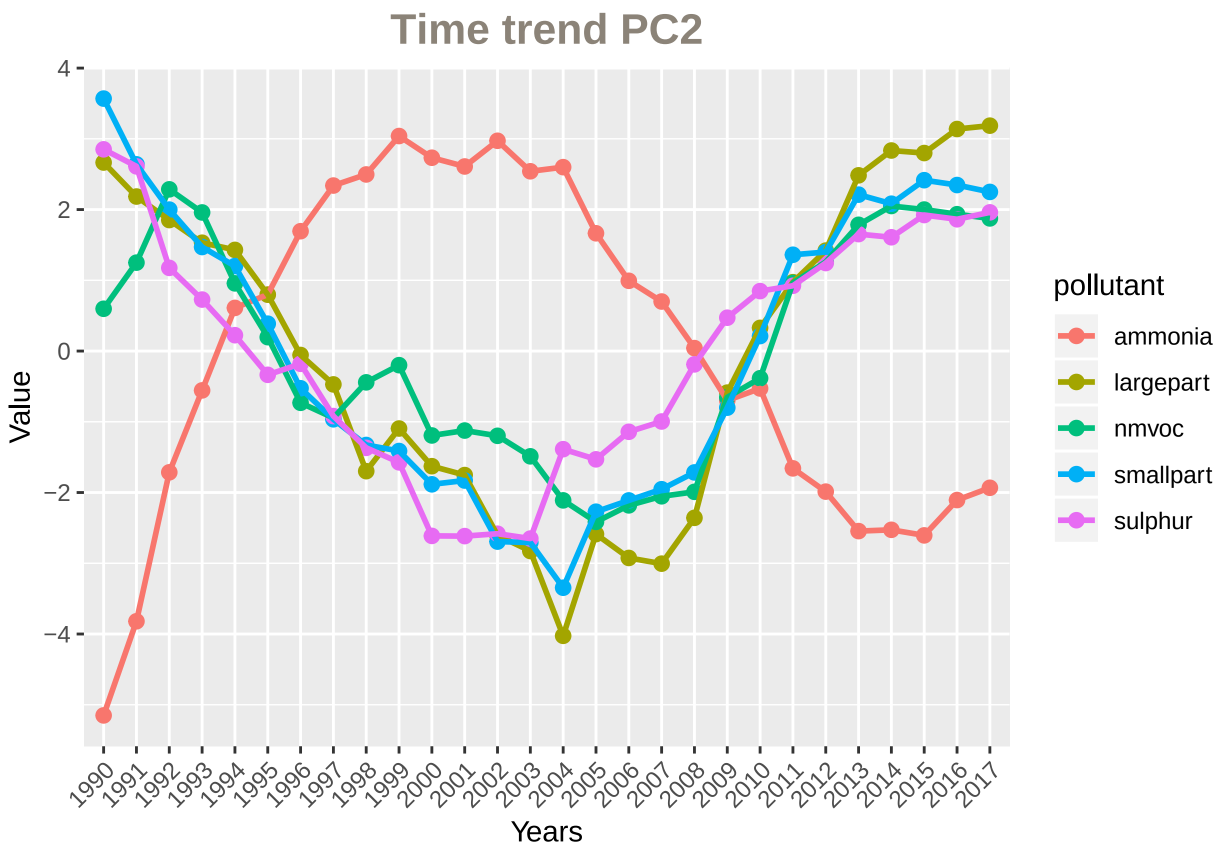
Here is the plot of the first principal component over time for the five pollutant in the dataset:

```
print(PC1_plot)
```



Here is the same plot for the second principal component:

```
print(PC2_plot)
```


Time trend PC2
Value
4
2
0
−2
−4
Years
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
pollutant
ammonia
largepart
nmvoc
smallpart
sulphur