Assignment 2

Group 4

2024-09-16

```
options(contrasts = c("contr.sum", "contr.poly"))
require("ggplot2")
require("dplyr")
require("ppcor")
require("caret")
require("tidyr")
require("stringr")
require("lubridate")
require("tsibble")
require("ggfortify")
require("gridExtra")

library(imputeTS)   # Time series missing value imputation

library(jsonlite)   # handle JSON data returned by Frost
library(tidyr)      # unpack data from JSON format
library(tidyverse)  # data manipulation with mutate etc, string formatting
library(lubridate)  # process date and time information
library(tsibble)    # special tibbles for time series
library(fpp3)       # autoplot() and gg_season() for time series
library(readr)      # to read the Frost client ID from file
```

Task 1: Dimension reduction on air quality data

Part A: Get

- Obtain data from https://archive.ics.uci.edu/dataset/360/air+quality.
- Provide a brief description of the data based on the information from the website.

```
airquality <- read.csv("AirQualityUCI.csv", sep=";")
summary(airquality)
```

```
##      Date               Time               CO.GT.           PT08.S1.CO.
##  Length:9471        Length:9471        Length:9471        Min.   :-200
##  Class :character   Class :character   Class :character   1st Qu.: 921
##  Mode  :character   Mode  :character   Mode  :character   Median :1053
##                                                           Mean   :1049
##                                                           3rd Qu.:1221
##                                                           Max.   :2040
##                                                           NA's   :114
##     NMHC.GT.         C6H6.GT.          PT08.S2.NMHC.       NOx.GT.
##  Min.   :-200.0   Length:9471        Min.   :-200.0   Min.   :-200.0
##  1st Qu.:-200.0   Class :character   1st Qu.: 711.0   1st Qu.:  50.0
```

```
##  Median :-200.0   Mode  :character   Median : 895.0   Median : 141.0
##  Mean   :-159.1                      Mean   : 894.6   Mean   : 168.6
##  3rd Qu.:-200.0                      3rd Qu.:1105.0   3rd Qu.: 284.0
##  Max.   :1189.0                      Max.   :2214.0   Max.   :1479.0
##  NA's   :114                         NA's   :114      NA's   :114
##   PT08.S3.NOx.     NO2.GT.         PT08.S4.NO2.   PT08.S5.O3.
##  Min.   :-200   Min.   :-200.00   Min.   :-200   Min.   :-200.0
##  1st Qu.: 637   1st Qu.:  53.00   1st Qu.:1185   1st Qu.: 700.0
##  Median : 794   Median :  96.00   Median :1446   Median : 942.0
##  Mean   : 795   Mean   :  58.15   Mean   :1391   Mean   : 975.1
##  3rd Qu.: 960   3rd Qu.: 133.00   3rd Qu.:1662   3rd Qu.:1255.0
##  Max.   :2683   Max.   : 340.00   Max.   :2775   Max.   :2523.0
##  NA's   :114    NA's   :114       NA's   :114    NA's   :114
##       T                  RH                  AH                X
##  Length:9471        Length:9471        Length:9471        Mode:logical
##  Class :character   Class :character   Class :character   NA's:9471
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##    X.1
##  Mode:logical
##  NA's:9471
##
##
##
##
##
```

```
sum(is.na(airquality))
```

```
## [1] 19854
```

```
datetime <- with(airquality, ymd(as.Date(airquality$Date,format="%Y/%m/%d")) + hms(gsub("\\.", ":", air
```

```
## Warning in .parse_hms(..., order = "HMS", quiet = quiet): Some strings failed
## to parse
```

```
airquality <- cbind(datetime = datetime, airquality)
airquality <- airquality %>% subset(select = -c(Time, Date, X, X.1))
airquality$CO.GT. <- as.integer(round(as.numeric(gsub("\\,", ".", airquality$CO.GT.)), digits=0))
airquality$PT08.S1.CO. <- as.numeric(airquality$PT08.S1.CO.)
airquality$C6H6.GT. <- as.numeric(gsub("\\,", ".", airquality$C6H6.GT.))
airquality$PT08.S2.NMHC. <- as.numeric(airquality$PT08.S2.NMHC.)
airquality$PT08.S3.NOx. <- as.numeric(airquality$PT08.S3.NOx.)
airquality$PT08.S4.NO2. <- as.numeric(airquality$PT08.S4.NO2.)
airquality$PT08.S5.O3. <- as.numeric(airquality$PT08.S5.O3.)
airquality$T <- as.numeric(gsub("\\,", ".", airquality$T))
airquality$RH <- as.numeric(gsub("\\,", ".", airquality$RH))
airquality$AH <- as.numeric(gsub("\\,", ".", airquality$AH))

airquality <- airquality %>%
  drop_na(datetime)

# Remove duplicates by single column (tsibble dosen't like duplicates)
```

```r
airquality <- airquality[!duplicated(airquality$datetime), ]

summary(airquality)
```

```
##     datetime                          CO.GT.         PT08.S1.CO.
##  Min.   :2001-01-20 00:00:00.00   Min.   :-200.00   Min.   :-200
##  1st Qu.:2008-08-20 05:45:00.00   1st Qu.:   1.00   1st Qu.: 915
##  Median :2016-03-20 11:30:00.00   Median :   2.00   Median :1048
##  Mean   :2016-03-26 06:26:13.15   Mean   : -36.23   Mean   :1042
##  3rd Qu.:2023-10-20 17:15:00.00   3rd Qu.:   3.00   3rd Qu.:1216
##  Max.   :2031-12-20 23:00:00.00   Max.   :  12.00   Max.   :2040
##     NMHC.GT.        C6H6.GT.        PT08.S2.NMHC.      NOx.GT.
##  Min.   :-200.0   Min.   :-200.000   Min.   :-200.0   Min.   :-200.0
##  1st Qu.:-200.0   1st Qu.:   4.100   1st Qu.: 714.8   1st Qu.:  44.0
##  Median :-200.0   Median :   8.000   Median : 898.0   Median : 133.0
##  Mean   :-156.3   Mean   :   1.434   Mean   : 895.7   Mean   : 160.6
##  3rd Qu.:-200.0   3rd Qu.:  13.725   3rd Qu.:1108.2   3rd Qu.: 275.0
##  Max.   :1189.0   Max.   :  63.700   Max.   :2214.0   Max.   :1479.0
##   PT08.S3.NOx.       NO2.GT.        PT08.S4.NO2.   PT08.S5.O3.
##  Min.   :-200.0   Min.   :-200.0   Min.   :-200   Min.   :-200.0
##  1st Qu.: 647.0   1st Qu.:  49.0   1st Qu.:1201   1st Qu.: 699.0
##  Median : 801.0   Median :  94.0   Median :1463   Median : 940.0
##  Mean   : 802.2   Mean   :  53.3   Mean   :1400   Mean   : 970.5
##  3rd Qu.: 971.0   3rd Qu.: 130.0   3rd Qu.:1675   3rd Qu.:1250.2
##  Max.   :2683.0   Max.   : 340.0   Max.   :2775   Max.   :2523.0
##        T                  RH               AH
##  Min.   :-200.000   Min.   :-200.00   Min.   :-200.0000
##  1st Qu.:  10.700   1st Qu.:  33.80   1st Qu.:   0.6826
##  Median :  17.400   Median :  48.30   Median :   0.9879
##  Mean   :   9.363   Mean   :  38.79   Mean   :  -7.3410
##  3rd Qu.:  24.500   3rd Qu.:  61.70   3rd Qu.:   1.3220
##  Max.   :  44.600   Max.   :  88.70   Max.   :   2.2310
```

```r
head(airquality)
```

```
##              datetime CO.GT. PT08.S1.CO. NMHC.GT. C6H6.GT. PT08.S2.NMHC.
## 1 2010-03-20 18:00:00      3        1360      150     11.9          1046
## 2 2010-03-20 19:00:00      2        1292      112      9.4           955
## 3 2010-03-20 20:00:00      2        1402       88      9.0           939
## 4 2010-03-20 21:00:00      2        1376       80      9.2           948
## 5 2010-03-20 22:00:00      2        1272       51      6.5           836
## 6 2010-03-20 23:00:00      1        1197       38      4.7           750
##   NOx.GT. PT08.S3.NOx. NO2.GT. PT08.S4.NO2. PT08.S5.O3.    T   RH     AH
## 1     166         1056     113         1692        1268 13.6 48.9 0.7578
## 2     103         1174      92         1559         972 13.3 47.7 0.7255
## 3     131         1140     114         1555        1074 11.9 54.0 0.7502
## 4     172         1092     122         1584        1203 11.0 60.0 0.7867
## 5     131         1205     116         1490        1110 11.2 59.6 0.7888
## 6      89         1337      96         1393         949 11.2 59.2 0.7848
```

Contains the responses of a gas multisensor device deployed on the field in an Italian city. Hourly responses averages are recorded along with gas concentrations references from a certified analyzer. Multivariate (15) and time series Has missing values.

Hints

- See options of `read.table()` for correct import

Part B: Import and Visualize

- Load the data and convert to tsibble.
  - Make sure dates and hours are converted into proper time objects
  - Remove incomplete days at beginning and end of data
- Plot the data as is, preferably as multiple panels in a single plot
- Describe the data. What is most striking?

```r
airqual <- as_tsibble(airquality, index = datetime)

head(airqual)
```

```
## # A tsibble: 6 x 14 [1h] <UTC>
##   datetime            CO.GT. PT08.S1.CO. NMHC.GT. C6H6.GT. PT08.S2.NMHC. NOx.GT.
##   <dttm>               <int>       <dbl>    <int>    <dbl>         <dbl>   <int>
## 1 2001-01-20 00:00:00   -200        1046     -200      4.2           724    -200
## 2 2001-01-20 01:00:00      2        1275     -200      8.8           930     215
## 3 2001-01-20 02:00:00      2        1173     -200      7.5           878     300
## 4 2001-01-20 03:00:00      3        1163     -200      7.6           881    -200
## 5 2001-01-20 04:00:00      2        1054     -200      5.6           791     253
## 6 2001-01-20 05:00:00      1        1004     -200      4.8           753     181
## # i 7 more variables: PT08.S3.NOx. <dbl>, NO2.GT. <int>, PT08.S4.NO2. <dbl>,
## #   PT08.S5.O3. <dbl>, T <dbl>, RH <dbl>, AH <dbl>
```

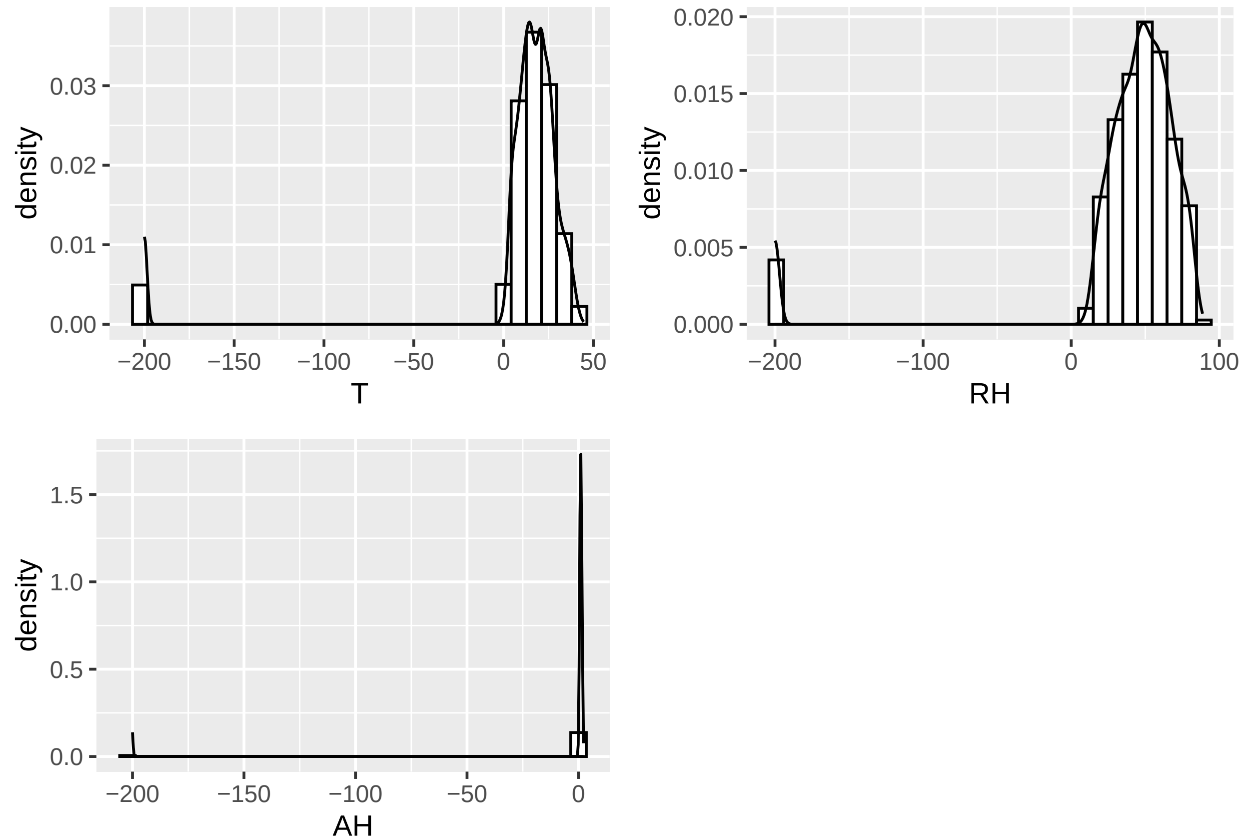
```r
tail(airqual)
```

```
## # A tsibble: 6 x 14 [1h] <UTC>
##   datetime            CO.GT. PT08.S1.CO. NMHC.GT. C6H6.GT. PT08.S2.NMHC. NOx.GT.
##   <dttm>               <int>       <dbl>    <int>    <dbl>         <dbl>   <int>
## 1 2031-12-20 18:00:00   -200         932     -200      6.1           817    -200
## 2 2031-12-20 19:00:00   -200         930     -200      5.3           781    -200
## 3 2031-12-20 20:00:00   -200         962     -200      5.3           780    -200
## 4 2031-12-20 21:00:00   -200         974     -200      5.5           790    -200
## 5 2031-12-20 22:00:00   -200        1055     -200      5.6           791    -200
## 6 2031-12-20 23:00:00   -200        1003     -200      4.6           744    -200
## # i 7 more variables: PT08.S3.NOx. <dbl>, NO2.GT. <int>, PT08.S4.NO2. <dbl>,
## #   PT08.S5.O3. <dbl>, T <dbl>, RH <dbl>, AH <dbl>
```

```r
#airqual %>%
#  autoplot(vars(CO.GT.))

hist_dens <- function(data, fact_n) {
  tmp <- data %>%
  ggplot(aes({{fact_n}})) +
  geom_histogram(aes(y = after_stat(density)), fill = "white", color="black") +
  stat_density(kernel = "gaussian", fill = NA, colour = "black")
  return(tmp)
}


p_T <- hist_dens(airqual, T)
p_RH <- hist_dens(airqual, RH)
```

```r
p_AH <- hist_dens(airqual, AH)

gridExtra::grid.arrange(p_T, p_RH, p_AH, ncol = 2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
# will crash
# airqual %>%
#   ggplot(aes(x=datetime, y=C6H6.GT.)) +
#   geom_line() +
#   geom_point() +
#   scale_x_continuous(breaks = seq(min(round(airqual$datetime)), max(round(airqual$datetime)), by = 2)
```

There seems to be multiple -200 in all the numerical (integer and Categorical) data that seems to be outliers. Should probably be removed from the dataset.

Part C: PCA of data as is

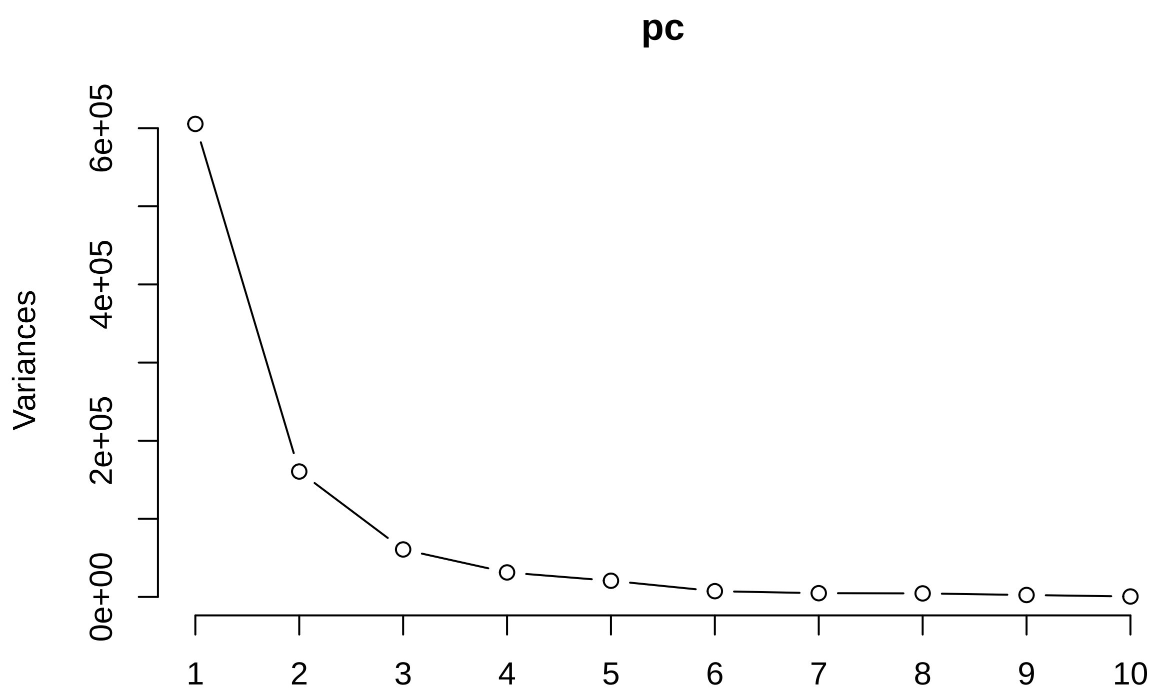
- Perform PCA on the data as prepared in B
- Create a screeplot and create biplots for 1st and 2nd and for 2nd and 3rd PCs
- Plot the scores for the PCs
- Comment on the results. Can you relate some features to your observations in part B?

```r
pc <- prcomp(airqual[,-1])
summary(pc)
```

```
## Importance of components:
##                             PC1      PC2       PC3       PC4       PC5      PC6
## Standard deviation     778.1140 400.6679 246.48348 177.04903 143.87039 85.91267
## Proportion of Variance   0.6737   0.1786   0.06761   0.03488   0.02303  0.00821
## Cumulative Proportion    0.6737   0.8524   0.91999   0.95487   0.97790  0.98612
##                             PC7      PC8      PC9     PC10     PC11    PC12
## Standard deviation     69.22065 67.23956 49.60452 23.66296 11.75704 2.33248
## Proportion of Variance  0.00533  0.00503  0.00274  0.00062  0.00015 0.00001
## Cumulative Proportion   0.99145  0.99648  0.99922  0.99984  0.99999 1.00000
##                          PC13
## Standard deviation     0.7655
## Proportion of Variance 0.0000
## Cumulative Proportion  1.0000
```

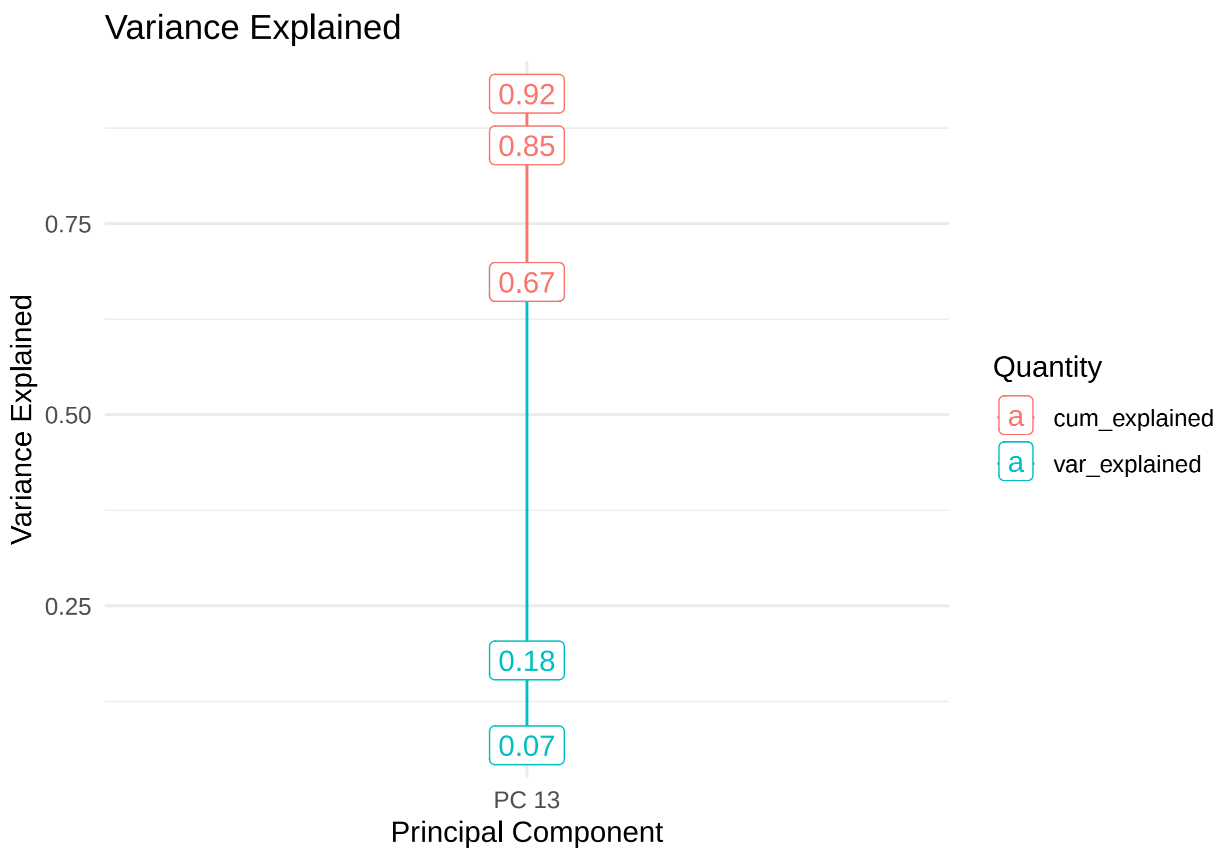
Screeplot

```r
plot(pc, type = "l")
```




```r
pc_v <- data.frame(PC = paste0("PC ", ncol(pc$x)),
                   var_explained = pc$sdev^2 / sum(pc$sdev^2)) %>%
          mutate(cum_explained = cumsum(var_explained))

pp <- pc_v[1:3,] %>%
  pivot_longer(!PC, names_to="Quantity", values_to="Explained") %>%
  ggplot(aes(x = PC, y = Explained, color=Quantity, group=Quantity))+
  geom_line() + geom_point() +
```

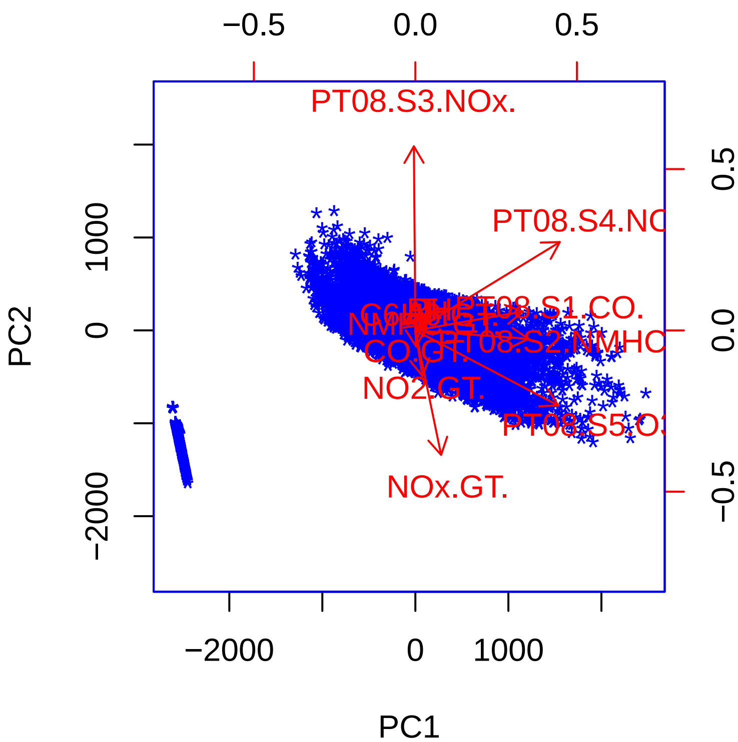
```r
  theme_minimal() +
  labs(title = "Variance Explained", x = "Principal Component",
       y = "Variance Explained")


pp + geom_label(aes(label = round(Explained, 2)))
```



0.91 of variance is explained in the first 3 principal components

Biplots

```r
biplot(pc, scale=0, col=c('blue', 'red'), xlabs=rep('*', nrow(pc$x[, 1:3])))
```



- Can clearly see the -200 outliers
- PRO8.S3.NOx positivly correlated between PC1 and PC2
- NOX.GT. opposite

Score plot

```r
d_pc <- data.frame(Time=airqual[,1], pc$x[,1:3])
head(d_pc)
```

```
##              datetime       PC1         PC2       PC3
## 1 2001-01-20 00:00:00 -271.0439    28.17748 -16.84424
## 2 2001-01-20 01:00:00  266.5296 -447.68869 237.54836
## 3 2001-01-20 02:00:00   52.8013 -366.42323 295.90569
## 4 2001-01-20 03:00:00  -41.8403   -43.85099 -58.31304
## 5 2001-01-20 04:00:00 -183.2791 -230.13360 289.41216
## 6 2001-01-20 05:00:00 -295.1182 -140.12075 264.80888
```

```r
# d_pc %>%
#   pivot_longer(!Time, names_to="Sensor", values_to="Measurement") %>%
#   ggplot(aes(x = Time, y = Measurement)) +
#   geom_line() +
#   geom_point() +
#   theme_minimal() +
#   facet_grid(Sensor ~.) +
#   xlab("Time [a.u.]")
```

Hints

- `ggfortify` provides `autoplot()` for PCA results for ggplot-style biplots
- To plot the scores, you can use the same code as for plotting the original data

Part D: Missing values

- Identify missing values in the time series
- Investigate to which degree missing values occur at the same time for multiple sensors
- Is one or are multiple sensors behaving peculiarly? How would you handle this?
- Discuss options for handling missing values: (a) drop all time points containing any missing value, (b) impute values for missing values. In case of (b) choose a method for imputation. Justify your decisions.
- At the end of this step, you should have a version of the data containing only valid values. Plot these data as in Part B.

```
#summary(airqual)
#sum(is.na(ariqual))
```

Hints

- Remember `imputeTS`
- You can apply its functions to an entire dataframe, will be done column-wise

Part E: PCA of cleaned data

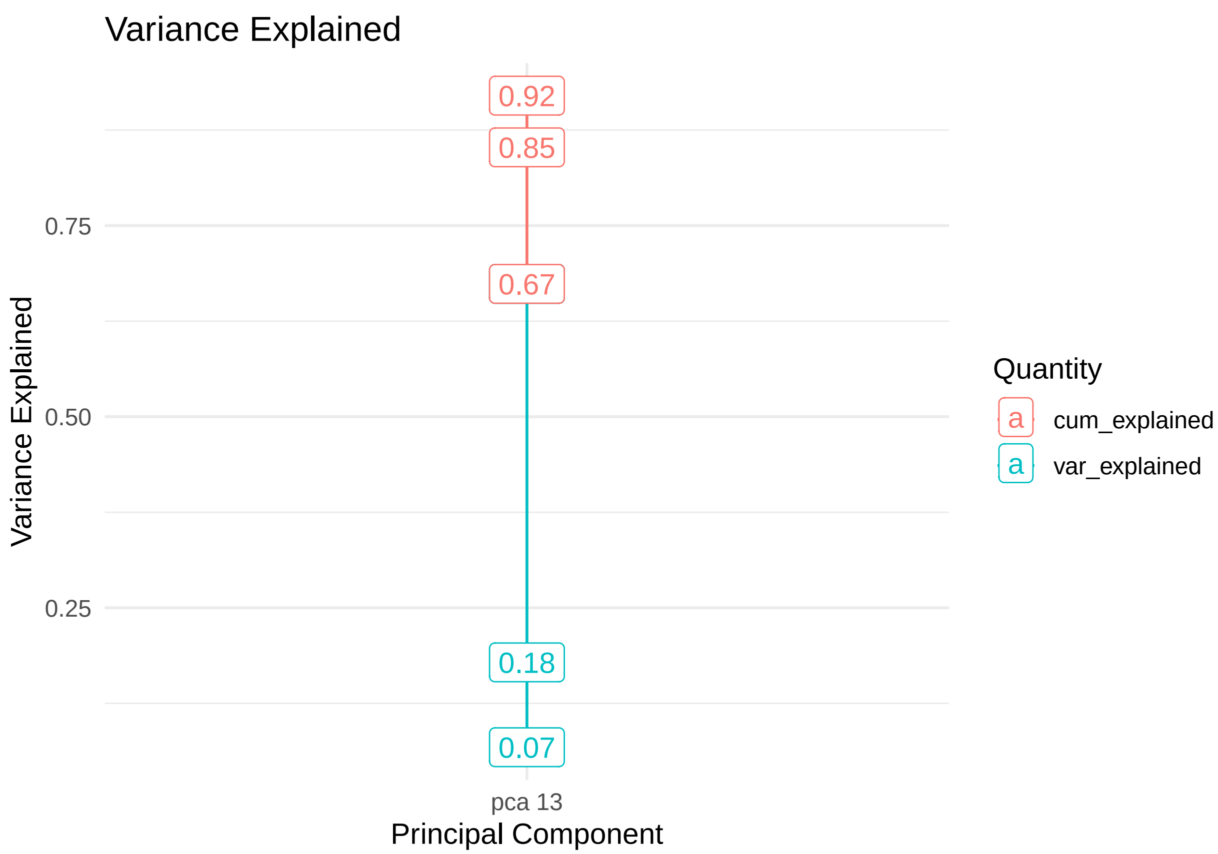
- Perform PCA on the data as prepared in D
- Create a screeplot and biplots for 1st/2nd, 2nd/3rd, 3rd/4th PC
- Compute total variance explained by 1st, 1st and 2nd, 1st to 3rd, … PCs
- Choose how many PCs to keep and transform data back to original sample space
- Plot the result against the cleaned data, compare and discuss
- Also plot the scores, zoom in to short time intervals and look at periodicity
- Can you interpret certain PCs?

```
clean_airqal <- airqual # Change later to proper

# screeplot
pca <- prcomp(clean_airqal[,-1])
summary(pca)
```

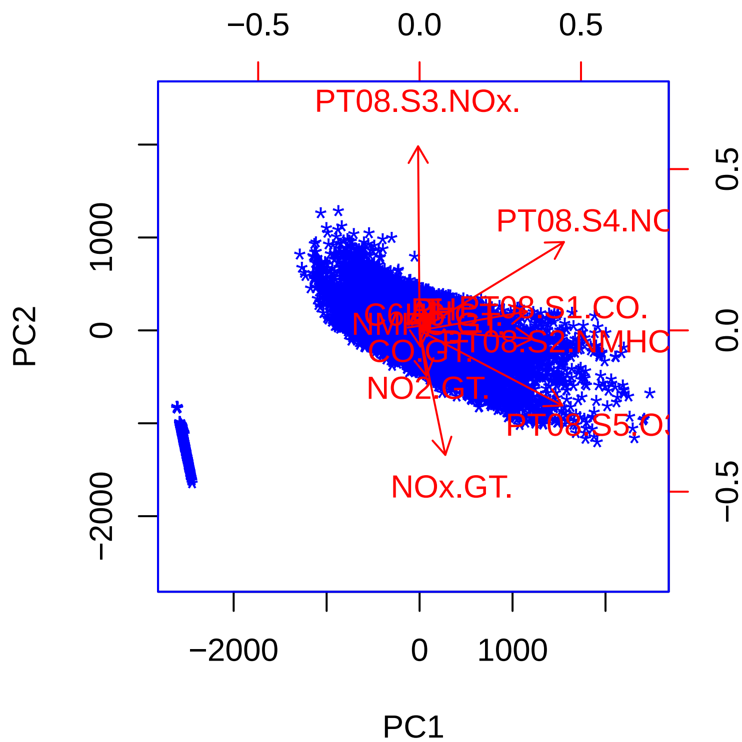
```
## Importance of components:
##                             PC1      PC2       PC3       PC4       PC5      PC6
## Standard deviation     778.1140 400.6679 246.48348 177.04903 143.87039 85.91267
## Proportion of Variance   0.6737   0.1786   0.06761   0.03488   0.02303  0.00821
## Cumulative Proportion    0.6737   0.8524   0.91999   0.95487   0.97790  0.98612
##                             PC7      PC8      PC9     PC10     PC11    PC12
## Standard deviation     69.22065 67.23956 49.60452 23.66296 11.75704 2.33248
## Proportion of Variance  0.00533  0.00503  0.00274  0.00062  0.00015 0.00001
## Cumulative Proportion   0.99145  0.99648  0.99922  0.99984  0.99999 1.00000
##                           PC13
## Standard deviation     0.7655
## Proportion of Variance 0.0000
## Cumulative Proportion  1.0000
```

```
pca_v <- data.frame(pca = paste0("pca ", ncol(pca$x)),
                   var_explained = pca$sdev^2 / sum(pca$sdev^2)) %>%
         mutate(cum_explained = cumsum(var_explained))
```

```r
pp <- pca_v[1:3,] %>%
  pivot_longer(!pca, names_to="Quantity", values_to="Explained") %>%
  ggplot(aes(x = pca, y = Explained, color=Quantity, group=Quantity))+
  geom_line() + geom_point() +
  theme_minimal() +
  labs(title = "Variance Explained", x = "Principal Component",
       y = "Variance Explained")


pp + geom_label(aes(label = round(Explained, 2)))
```



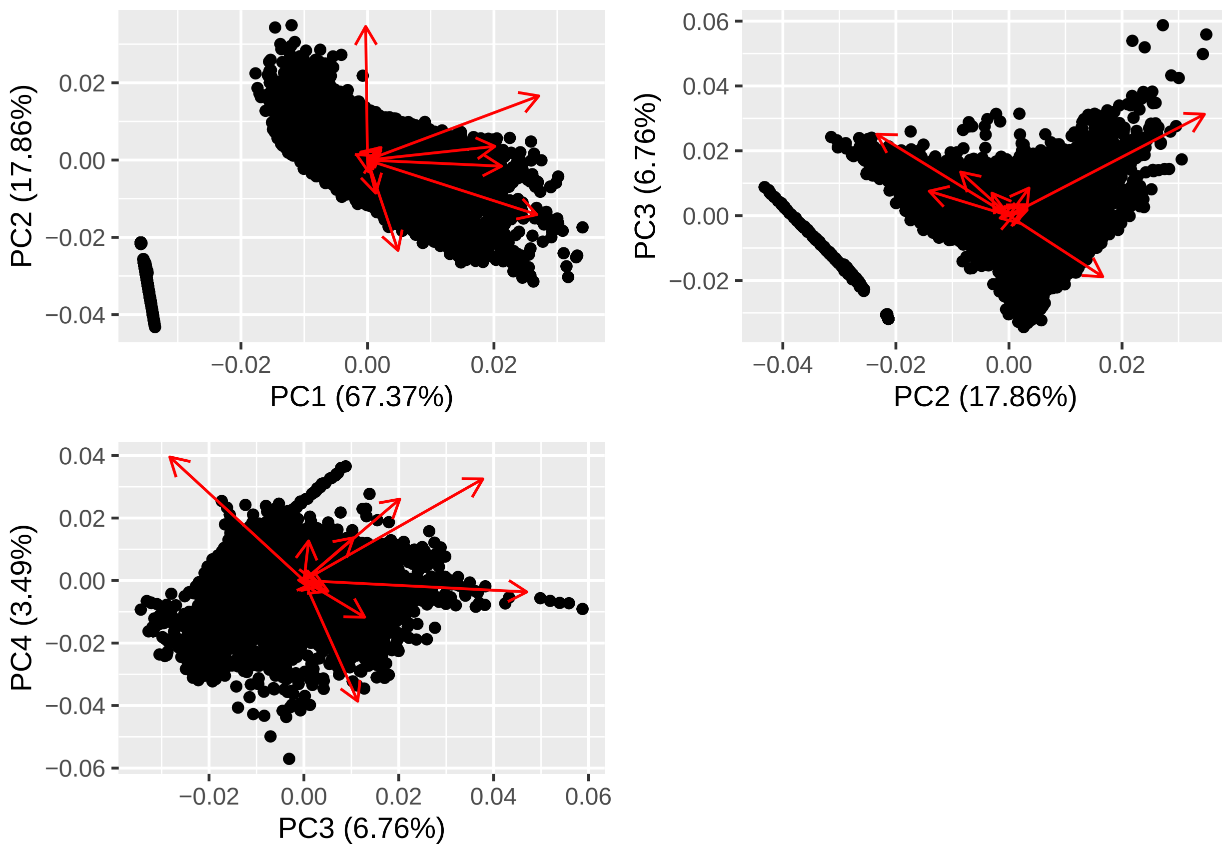

```r
# biplots
biplot(pca, scale=0, col=c('blue', 'red'), xlabs=rep('*', nrow(pca$x[, 1:3])))
```




```
head(clean_airqal[-1])
```

```
## # A tibble: 6 x 13
##   CO.GT. PT08.S1.CO. NMHC.GT. C6H6.GT. PT08.S2.NMHC. NOx.GT. PT08.S3.NOx.
##    <int>       <dbl>    <int>    <dbl>         <dbl>   <int>        <dbl>
## 1   -200        1046     -200      4.2           724    -200          848
## 2      2        1275     -200      8.8           930     215          649
## 3      2        1173     -200      7.5           878     300          738
## 4      3        1163     -200      7.6           881    -200          748
## 5      2        1054     -200      5.6           791     253          830
## 6      1        1004     -200      4.8           753     181          879
## # i 6 more variables: NO2.GT. <int>, PT08.S4.NO2. <dbl>, PT08.S5.O3. <dbl>,
## #   T <dbl>, RH <dbl>, AH <dbl>
```

```
bp1 <- autoplot(pca, data = clean_airqal[,-1], loadings = T, loadings.labels = T, loadings.label.size =
bp2 <- autoplot(pca, data = clean_airqal[,-1], loadings = T, loadings.labels = T, loadings.label.size =
bp3 <- autoplot(pca, data = clean_airqal[,-1], loadings = T, loadings.labels = T, loadings.label.size =
gridExtra::grid.arrange(bp1, bp2, bp3, ncol = 2)
```



Total variance

```r
#Variance explained
summary(pca)$importance[3,]
```

```
##     PC1     PC2     PC3     PC4     PC5     PC6     PC7     PC8     PC9    PC10
## 0.67374 0.85238 0.91999 0.95487 0.97790 0.98612 0.99145 0.99648 0.99922 0.99984
##    PC11    PC12    PC13
## 0.99999 1.00000 1.00000
```

How much to keep?

Keep to 0.95 mark, so PC1 to PC4

```r
#cut_pc <- pc[,1:4]
```

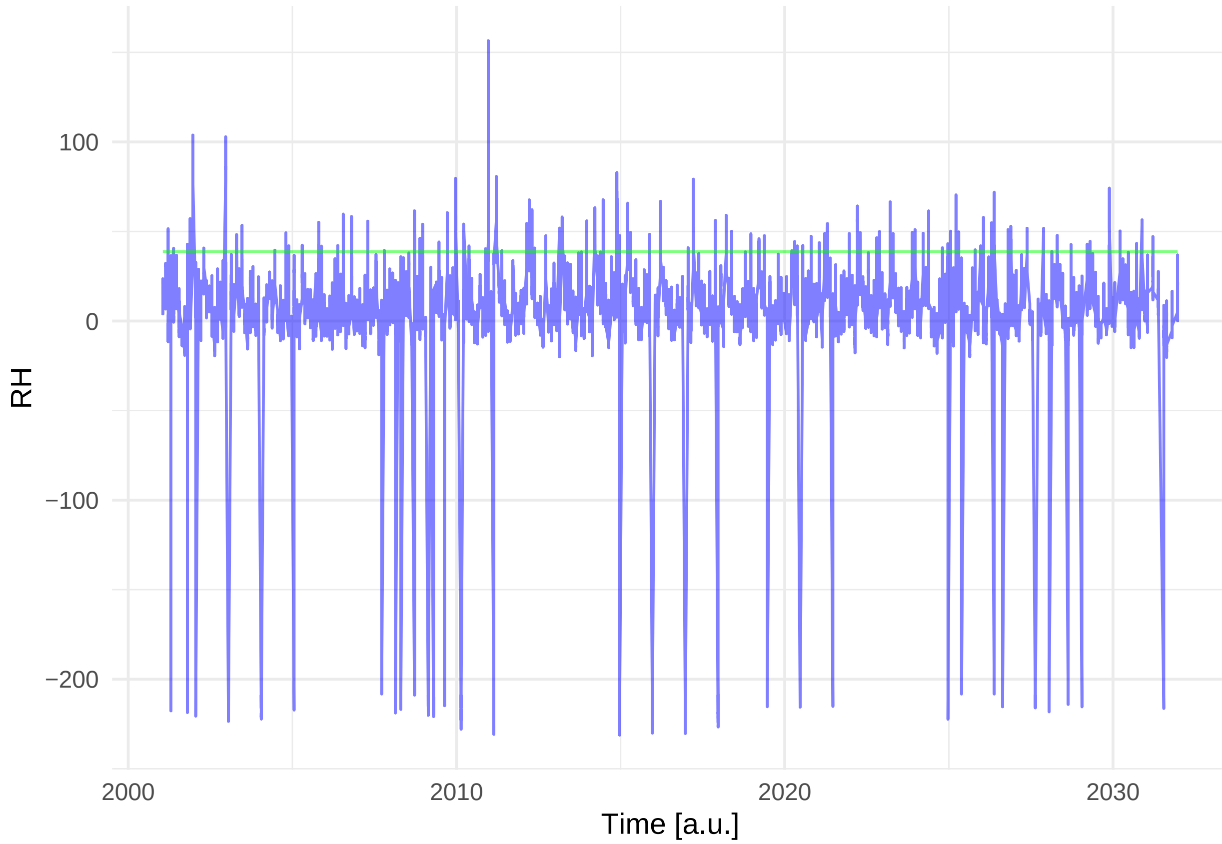
Transform back

```r
# Need to substract means back and rescale the variables
#t <- predict(pc)
#airqual_pc <- t(t(pc$x %*% t(pc$rotation)) * pc$scale + pc$center)
#airqual_pc
t <- datetime
x_1 <- pca$x[, 1:4] %*% t(pca$rotation[, 1:4])
x_2 <- t(pca$center + pca$scale * t(x_1))
```

Plot and discuss

```
ggplot() +
  geom_line(data = data.frame(datetime = clean_airqal$datetime, x_1), aes(datetime, RH), color = "blue"
  geom_line(data = data.frame(datetime = clean_airqal$datetime, x_2), aes(datetime, RH), color = "green
  geom_line(data = clean_airqal[c("datetime", "RH"),] , aes(datetime, RH), color = "red") +
  theme_minimal() +
  xlab("Time [a.u.]")
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

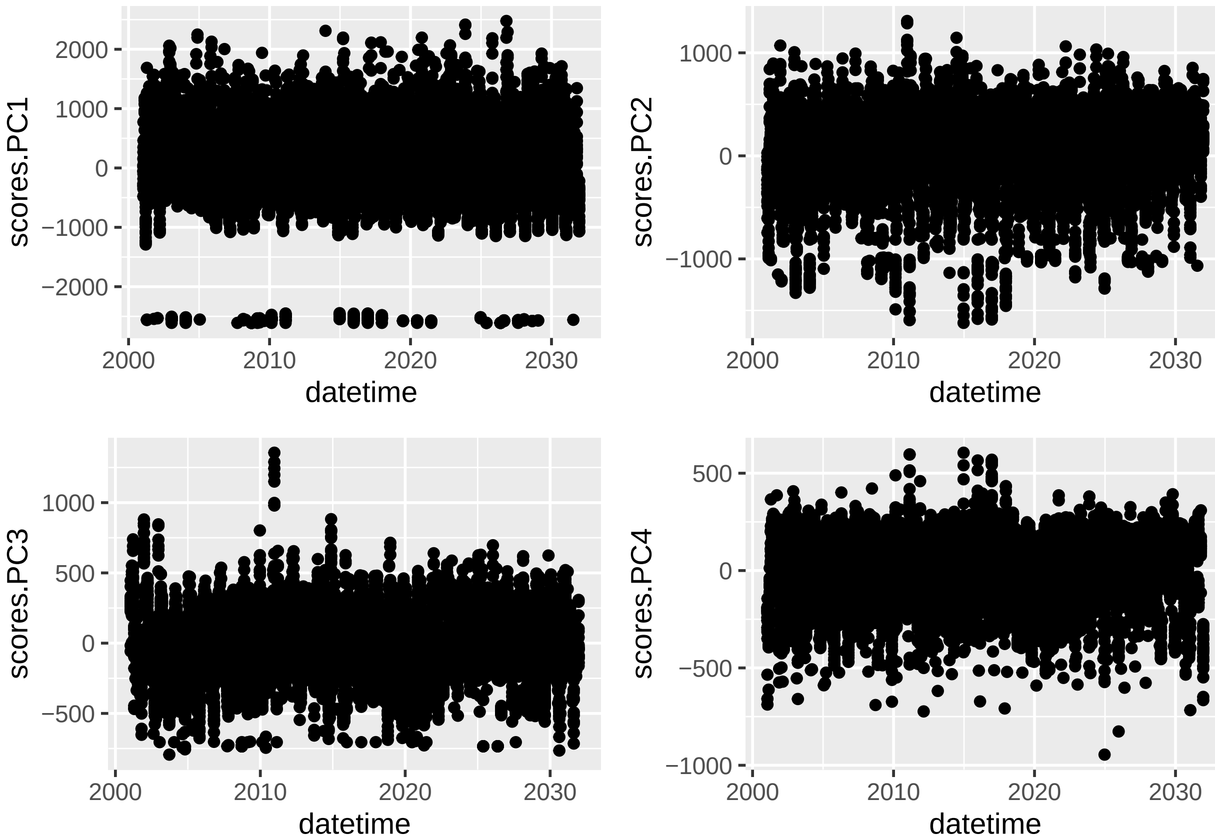


Looks mostly the same, but deviates slightly as we removed all afer PC4.

Plot the scores

```
scores_time <- data.frame(datetime = clean_airqal$datetime, scores = pca$x)

p1 <- scores_time %>%
  ggplot(aes(datetime, scores.PC1)) +
  geom_point()
p2 <- scores_time %>%
  ggplot(aes(datetime, scores.PC2)) +
  geom_point()
p3 <- scores_time %>%
  ggplot(aes(datetime, scores.PC3)) +
  geom_point()
```

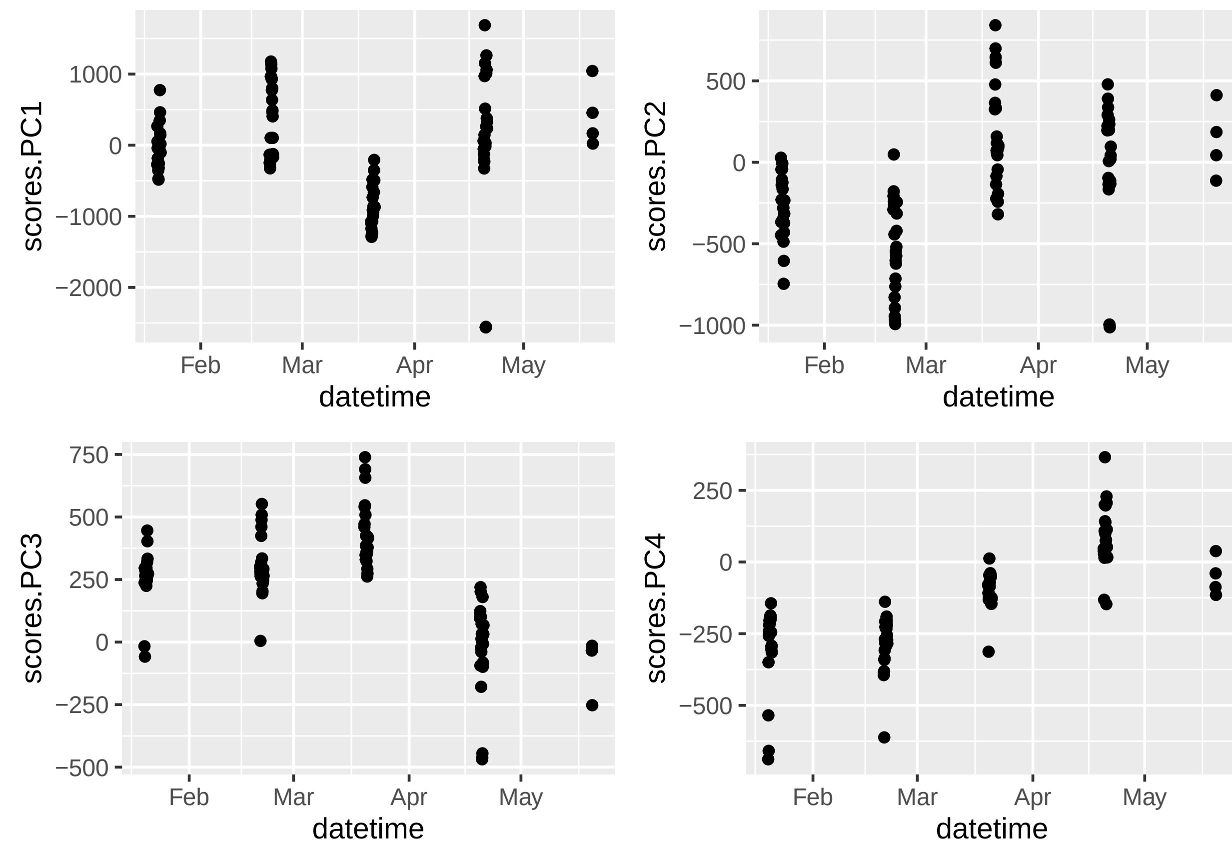
```r
p4 <- scores_time %>%
  ggplot(aes(datetime, scores.PC4)) +
  geom_point()

gridExtra::grid.arrange(p1, p2, p3, p4, ncol = 2)
```



Whole

Zoomed

```r
scores_time <- data.frame(datetime = clean_airqal$datetime[1:100], scores = pca$x[1:100,])

p1 <- scores_time %>%
  ggplot(aes(datetime, scores.PC1)) +
  geom_point()
p2 <- scores_time %>%
  ggplot(aes(datetime, scores.PC2)) +
  geom_point()
p3 <- scores_time %>%
  ggplot(aes(datetime, scores.PC3)) +
  geom_point()
p4 <- scores_time %>%
  ggplot(aes(datetime, scores.PC4)) +
  geom_point()

gridExtra::grid.arrange(p1, p2, p3, p4, ncol = 2)
```



Task 2: STL and correlation on weather data

Part A: Data collection for a single station

Based on material from the lectures, write an R function that can obtain a daily average temperature series for a meteorological station from the Norwegian Met Institute's Frost service. The function shall return a tsibble.

Part B: Data preparation for a single station

- Identify gaps in the time series.
- Assume that gaps up to 31 days are acceptable. Find the earliest date in the time series such that all following data have no gaps longer than 31 days. Limit the time series to this.
- Create a regular time series by filling gaps in the tsibble with n/a-s.
- Impute values for the n/a-s. Justify your choice of imputation method.
- You should now have a regular time series with only numeric values.
- Remove all data for 29 February so all years have data for exactly 365 days.
- Combine all this code into a function for re-use later. The function should receive the original tsibble from part A as input and return a new tsibble.

Hints

- tidyverse provides functions such as has_gaps() and count_gaps()

Part C: Exploratory analysis for a single station

- Plot the temperature data as function of time
- Create density plots of original data and data with imputed values
- Turn the temperature data into a timeseries (ts) object
- Plot the autocorrelation function for lags up to 5.5 years; describe and discuss your observations
- Also plot the ACF only for short lags, up to four weeks
- Select some days distributed throughout the year and plot temperature as function of year for, e.g., 1 October, as a scatter plot. This plot can be useful to choose the seasonality window later (see Figs 7 and 8 in Cleveland et al, 1990)

Part D: STL analysis

- Perform STL on the data. Explore different values for the seasonality and trend windows (remember that we want to look at trends over many years!), the choice between robust STL or not, and possibly the lowpass filter window. Describe your observations. It might be interesting to look at the ACF of the remainder in the STL result.
- Consult the original STL paper by Cleveland et al. (1990) for suggestions on how to choose STL parameters.
- Based on your analysis, can you suggest a set of STL parameters to use for further work?

Part E: Multiple station analysis

- Obtain data from eight more stations. Two should be in the same part of Norway as the station from part A; then choose three stations each from two other parts of Norway. Data should cover several decades at least, so look for stations with long series.
- Preprocess the data as described in Part B. Find the latest starting date of any series and create a multivariate time series with data from all nine stations starting at this date.
- Obtain the cross-correlation matrix between the nine stations. Is there any structure in this 9x9 matrix?
- Perform STL individually on each of the nine stations using the parameters from part D. Compare the resulting trends. Are all STL results of equal quality?

Hints

You can get a list of all available stations from Frost using

```
#.stations_url = str_glue("https://{.client_id}@frost.met.no/sources/v0.jsonld")
#raw_stations <- fromJSON(URLencode(.stations_url), flatten=TRUE)
```

To limit this to stations with actual data, starting at least as early as 1950, coming from only some parts of Norway relevant columns, and limiting to relevant columns, filter the raw data as

```
# COUNTYS = c(fylke1, fylke2, etc)   # replace with names of "fylker" you are interested in
#
# stations <- unnest(raw_stations$data, cols='id') |>
#   select(id, validFrom, country, county, municipality, name, masl, `@type`) |>
#   mutate(validFrom=as.Date(validFrom)) |>
#   filter(`@type` == "SensorSystem" & validFrom <= "1950-01-01" & country == "Norge" & county %in% COU
```

Part F (bonus): PCA

- Perform PCA on the multivariate time series.