EM EDX initial data exploration

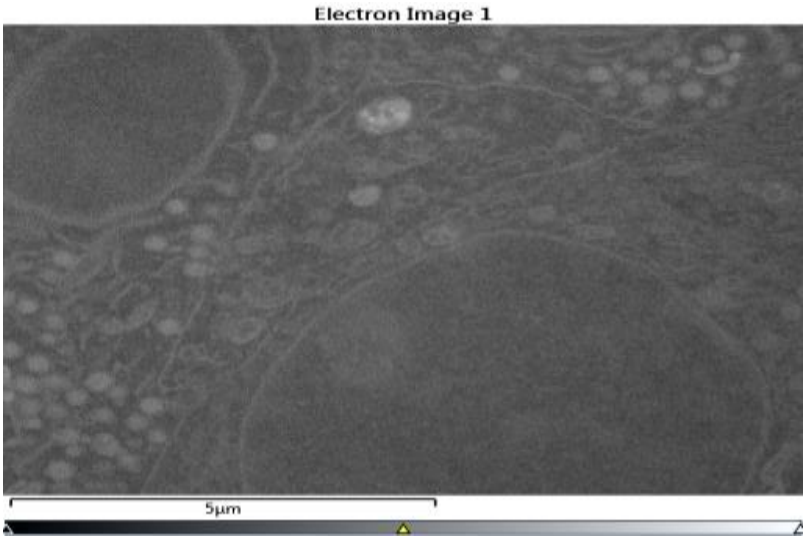
Data description

Data was provided by Drs Jeroen Kuipers of the UMCG. Exploratory research based on results presented in www.nature.com/scientificreports/: Scientific RepoRts | 7:45970

First data obtained concerns an EM image of the pancreas, with corresponding EDX data (X-ray detection).

```
Label:  EDS Map Data 1
Collected:  20/01/2017 15:04:15
Resolution (Width): 1024 pixels
Resolution (Height):    832 pixels
Map Width:  9.43µm
Map Height: 7.66µm
Accelerating Voltage:   20.0kV
Magnification:  31940x
Working Distance:   5.5mm
Number of Completed Frames: 15
Energy Range (keV): 20 keV
Number Of Channels: 2048
```

This is the EM picture:



EM picture

I got the corresponding intensity files (these are NOT the raw data files!) for these elements:

```
list.files(pattern = "*.csv")
```

```
## [1] "Cl Kα1.csv"           "Electron Image 1.csv" "Electron Image 2.csv"
## [4] "Fe Kα1.csv"           "N Kα1.csv"            "O Kα1.csv"
## [7] "Os Lα1.csv"           "P Kα1.csv"            "S Kα1.csv"
```

Combining the data

I will first attempt to get a single data file containing the intensities. There are data files for 7 elements and one for the tiff image. All are (supposedly) `1024 * 832` = 8.5196810^{5} pixels in size, so the resulting data matrix will have 851968 rows and 10 columns (one also for x and y position of the pixel).

I will start out with combining the data into one dataframe.

```
#atoms, sorted from low to high Mw
atoms <- c("N", "O", "P", "S", "Cl", "Fe", "Os")
base.names <- c(rep(" Kα1", 6), " Lα1")
file.extension <- ".csv"

#read EM picture as intensities csv file
em.intensities <- read.table("Electron Image 1.csv", header = F, sep=",")
##exporting has yielded a last column that only contains NAs - remove these
em.intensities <- em.intensities[, 1 : (ncol(em.intensities) - 1)]

#create dataframe based on em picture dimensions and populate with x & y
coordinates
nrows <- dim(em.intensities)[1]
ncols <- dim(em.intensities)[2]
x.coord <- rep(1 : nrows, each = ncols)
y.coord <- rep(1 :ncols, nrows)
dat <- data.frame(x = x.coord, y = y.coord)

#attach the em picture data
attach.as.column(em.intensities, "EM")

#read atom intensity for all files
for (i in 1:length(atoms)) {
  file.name <- paste(atoms[i], base.names[i], file.extension, sep = "")
  atom.intensity <- read.table(file.name, sep=",", head=F)
  #erroneous data format has trainling comma on each line
  atom.intensity <- atom.intensity[, 1:(ncol(atom.intensity)-1)]

  attach.as.column(atom.intensity, atoms[i])
}

head(dat)
```

```
##   x y    EM N O P S Cl Fe Os
## 1 1 1 10062 1 0 0 0  0  0  0
## 2 1 2 12119 1 1 0 0  0  0  0
```

```
## 3 1 3 10729 0 6 0 0  1  0  0
## 4 1 4 13112 0 0 1 0  0  0  0
## 5 1 5 10551 0 2 1 0  0  0  0
## 6 1 6 11540 0 3 0 0  1  0  0
```

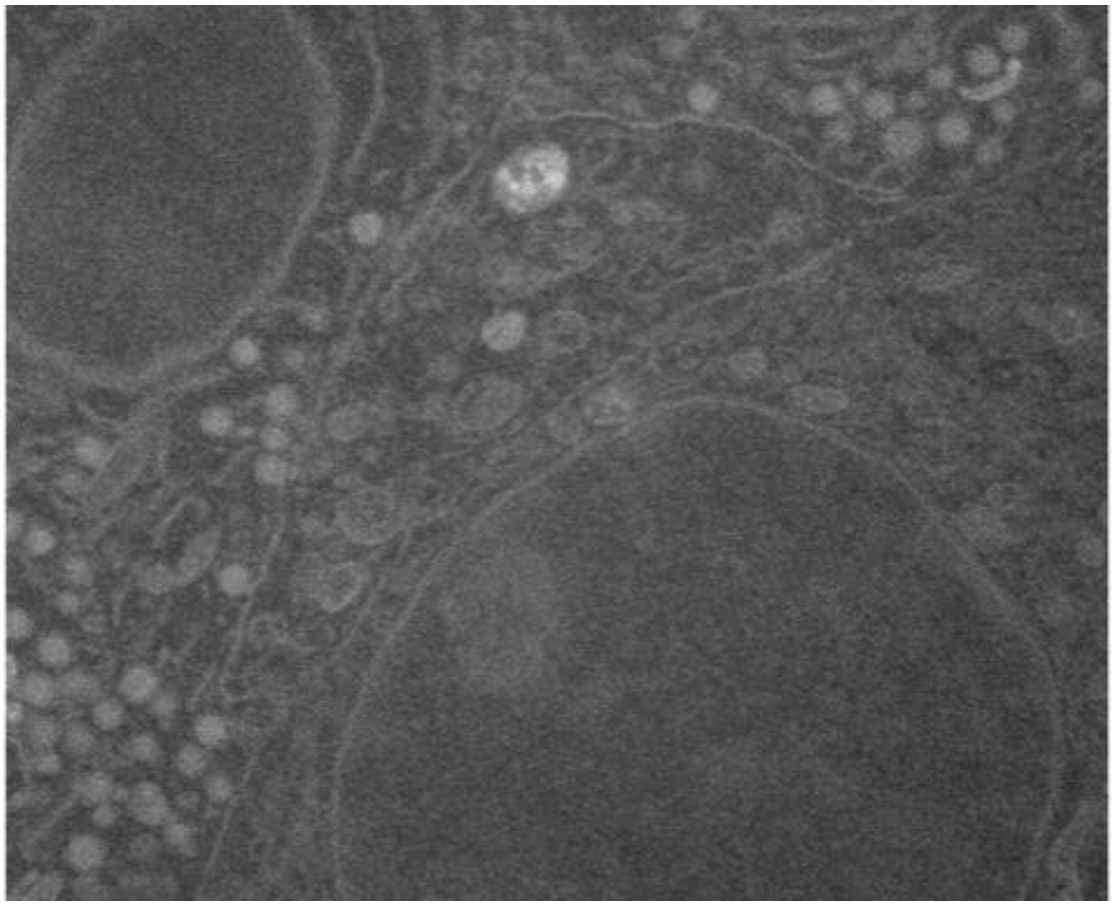
```
atoms.only <- dat[, atoms]
head(atoms.only)
```

```
##   N O P S Cl Fe Os
## 1 1 0 0 0  0  0  0
## 2 1 1 0 0  0  0  0
## 3 0 6 0 0  1  0  0
## 4 0 0 1 0  0  0  0
## 5 0 2 1 0  0  0  0
## 6 0 3 0 0  1  0  0
```

Initial exploration

Let's see whether I can get the EM image back. When the min-max scaling is applied, this seems like a rather good approximation of the original tiff image inserted above.

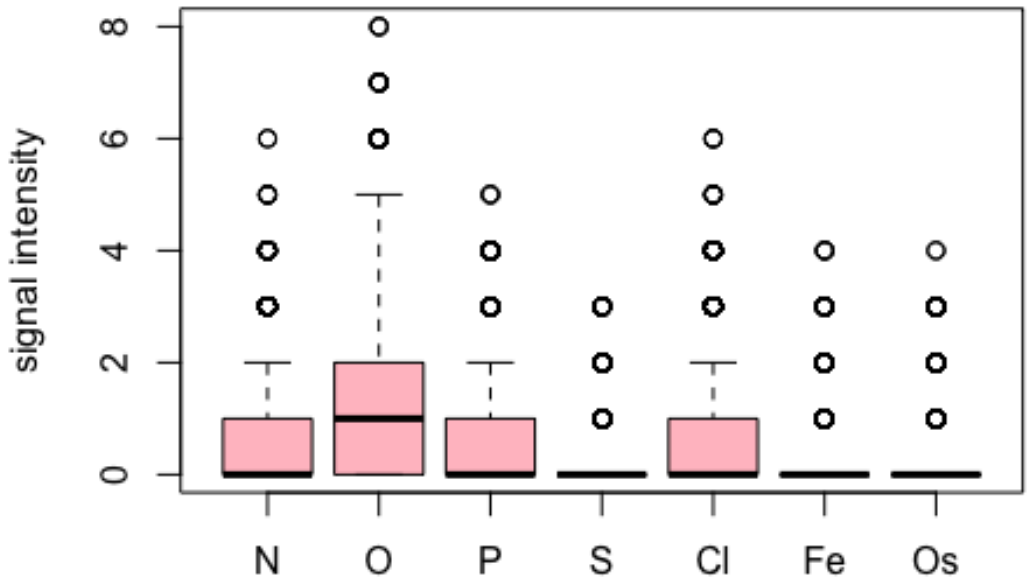
```
library(EBImage)
img <- Image(min.max.scale(t(as.matrix(em.intensities))))
display(img, method = "raster")
```



```
#dev.print(jpeg, filename="test.jpg", width=1024, height=832)
```

The intensities of the (non-EM) columns have a distribution like this

```
#for speed only a part
boxplot(atoms.only[1:100000, ], ylab = "signal intensity", col = rgb(1, 0,
0.2, 0.3))
```



This looks rather skewed, but is explained by the fact that a vast majority is probably zero intensity. Let's verify that.

```
apply(atoms.only, MARGIN = 2, FUN = function(x){
  count.zero <- sum(x==0)
  c("zeros" = count.zero, "perc" = round(count.zero/length(x)*100))
})
```

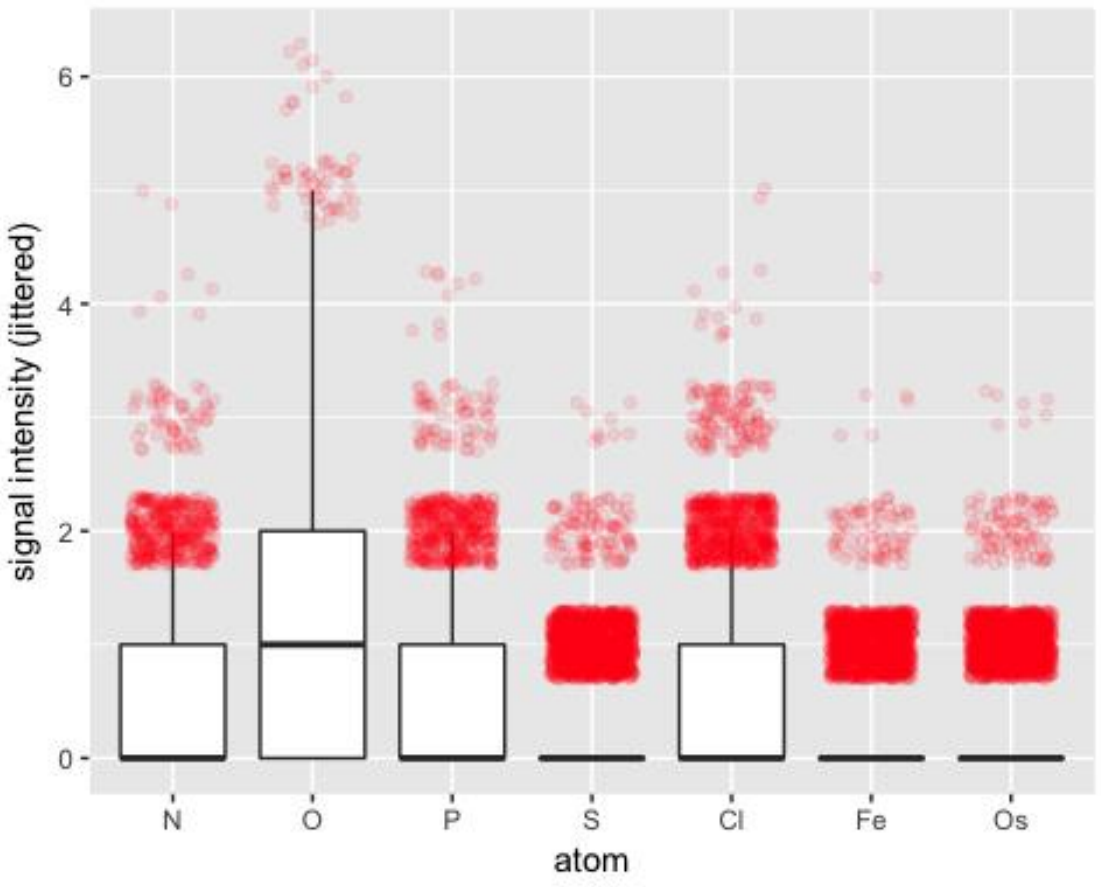
```
##            N      O      P      S     Cl     Fe     Os
## zeros 613623 267097 633058 750251 565772 758555 764071
## perc      72     31     74     88     66     89     90
```

So that assumption is correct. Not surprisingly, zero-counts are highest for S, Fe and Os, amd lowest for O.

The plot below shows that tehre are actually many outliers

```
library(ggplot2)
library(dplyr)
library(reshape)

tmp <- atoms.only[1:10000, ]
tmp <- melt(tmp) %>%
  group_by(variable) %>%
```

```
  mutate(outlier = value > median(value) + IQR(value) * 1.5) %>%
  ungroup

## Using  as id variables

ggplot(tmp, aes(variable, value)) +
  geom_boxplot(outlier.shape = NA) +
  geom_point(data = function(x) dplyr::filter_(x, ~ outlier), position =
position_jitter(height = 0.3, width = 0.3), colour = rgb(1, 0, 0.1, 0.1)) +
  labs(x = "atom", y = "signal intensity (jittered)")
```



Now have a look at basic correlations between these elements:

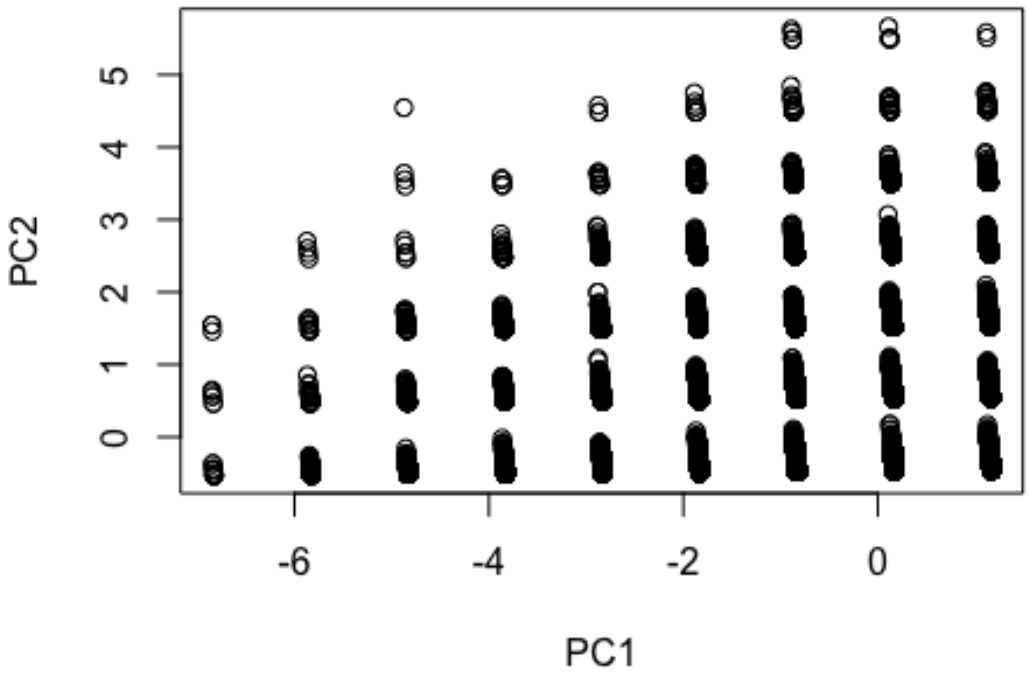
```
round(cor(atoms.only), 2)

##       N    O    P    S   Cl   Fe   Os
## N  1.00 0.01 0.02 0.01 0.02 0.01 0.01
## O  0.01 1.00 0.01 0.00 0.01 0.00 0.00
## P  0.02 0.01 1.00 0.01 0.02 0.01 0.02
## S  0.01 0.00 0.01 1.00 0.01 0.00 0.01
## Cl 0.02 0.01 0.02 0.01 1.00 0.01 0.02
## Fe 0.01 0.00 0.01 0.00 0.01 1.00 0.01
## Os 0.01 0.00 0.02 0.01 0.02 0.01 1.00
```

These correlations are of course really low. But what is the situation if combinations of (0, 0) are removed? They are the vast majority and probably disturb the picture.

PCA

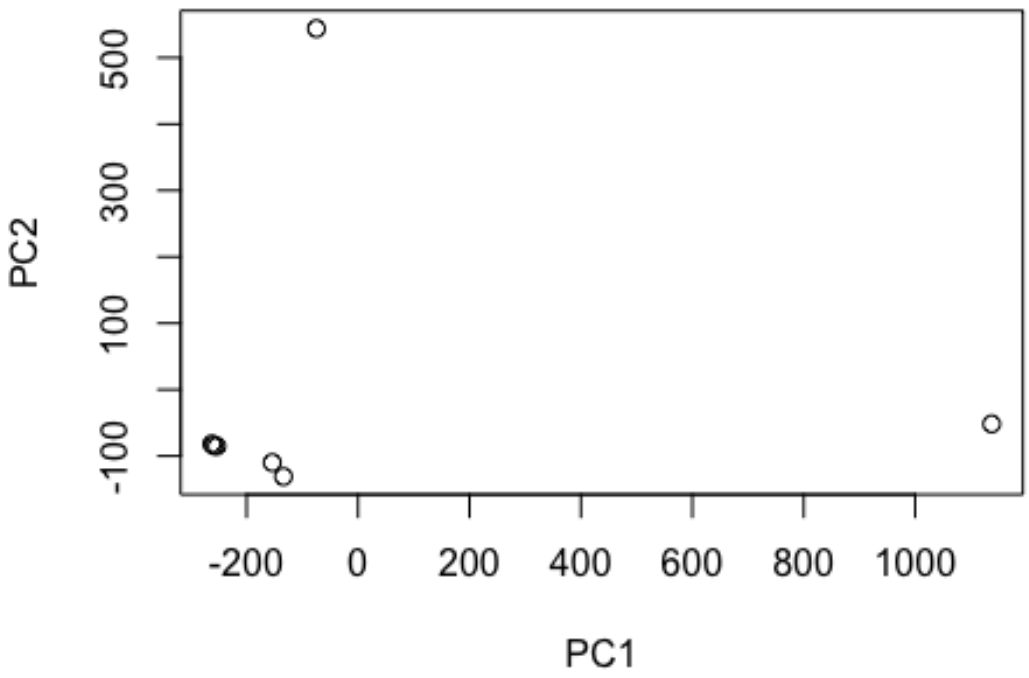
Perform PCS by rows

```
atomsPCAcols <- prcomp(atoms.only)
plot(atomsPCAcols$x[,1:2])
```



PCA by columns

```
atomsPCArows <- prcomp(t(atoms.only))
plot(atomsPCArows$x[,1:2])
```

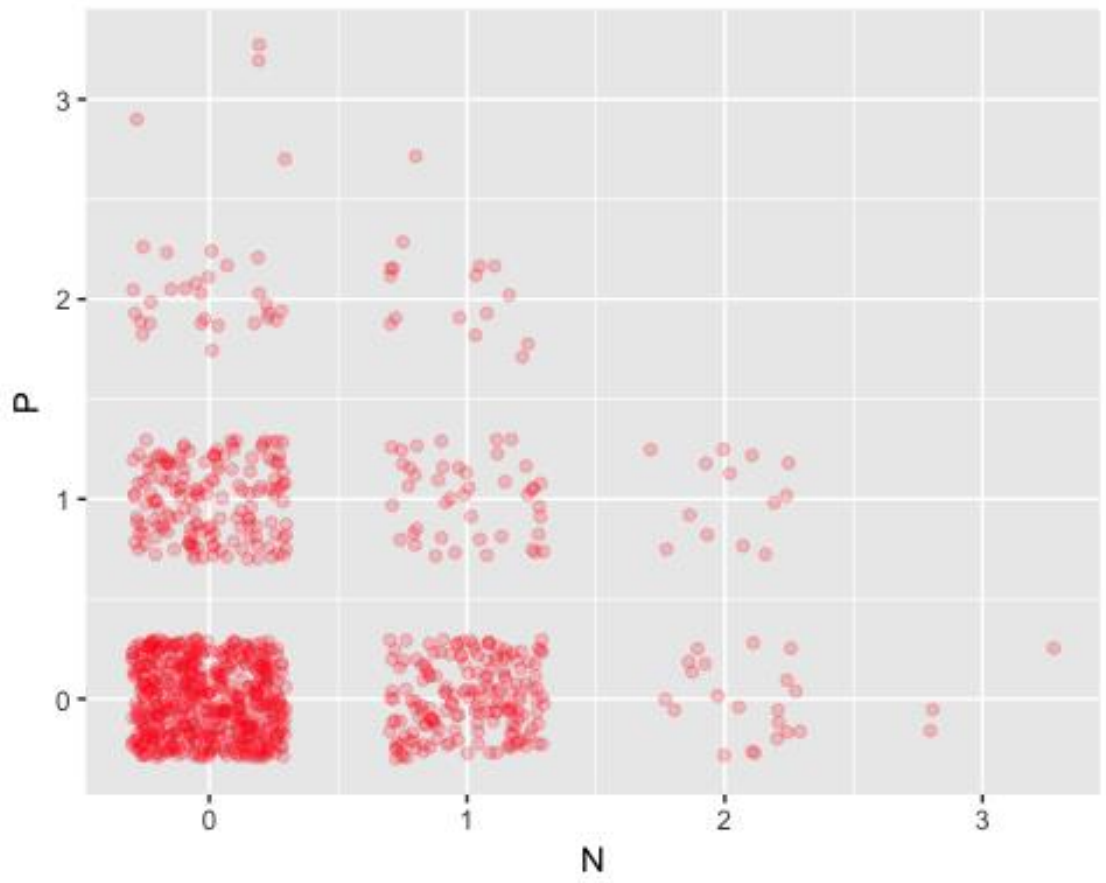


since this looks rather dodgy, I will no first try to reproduce the pictures as shown in the Word document "proces times_Site 1_2017-09-20_17-08-54.docx".

K-Means clustering

I will attempt to do a k-Means clustering on this dataset in order to find out whether patterns appear.

```
#library(ggplot2)
set.seed(1)
plot.sample <- atoms.only[sample.int(nrow(atoms.only), 1000), ]
ggplot(plot.sample, aes(N, P)) +
  geom_point(position = position_jitter(height = 0.3, width = 0.3),
             colour = rgb(1, 0, 0.1, 0.2))
```



So, indeed there does not seem correlation between atom measurements at first glance - it is not shown but this was also tested for some other combinations

```
set.seed(1)
kmeans.clusters <- kmeans(plot.sample, centers = 10, nstart = 20)
kmeans.clusters$withinss
```

```
##  [1]  61.95968  31.09091 117.56250 172.85256  53.00000 129.14394  77.97714
##  [8]  81.78862 105.50515  69.04545
```

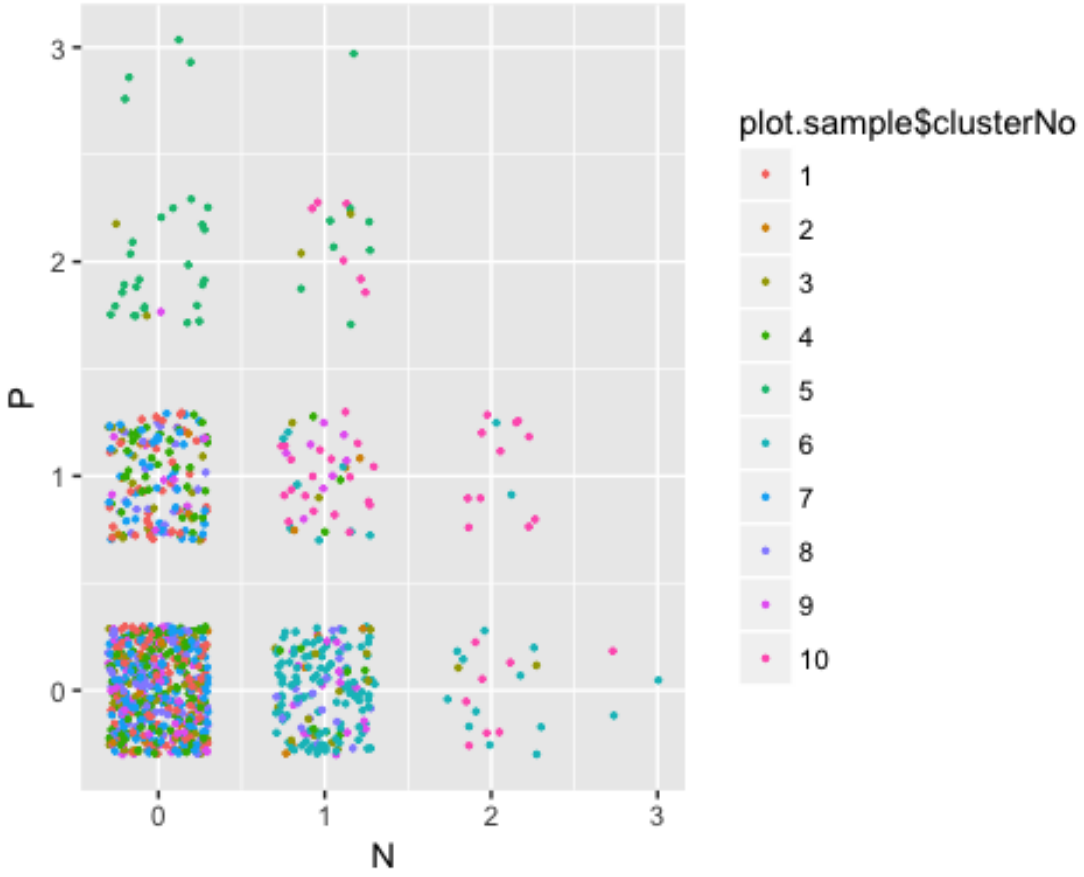
```
kmeans.clusters$totss
```

```
## [1] 2543.847
```

```
kmeans.clusters$centers
```

```
##             N          O          P          S        Cl         Fe
## 1  0.00000000 0.00000000 0.27419355 0.17741935 0.0000000 0.00000000
## 2  0.24242424 0.03030303 0.24242424 0.09090909 0.3636364 1.09090909
## 3  0.26250000 2.18750000 0.32500000 0.15000000 1.4125000 0.20000000
## 4  0.07051282 0.48076923 0.24358974 0.10897436 1.3974359 0.06410256
## 5  0.22222222 0.80555556 2.13888889 0.11111111 0.4166667 0.02777778
## 6  1.12878788 0.56818182 0.07575758 0.13636364 0.2196970 0.09848485
## 7  0.00000000 1.00000000 0.21142857 0.05714286 0.0000000 0.09142857
```

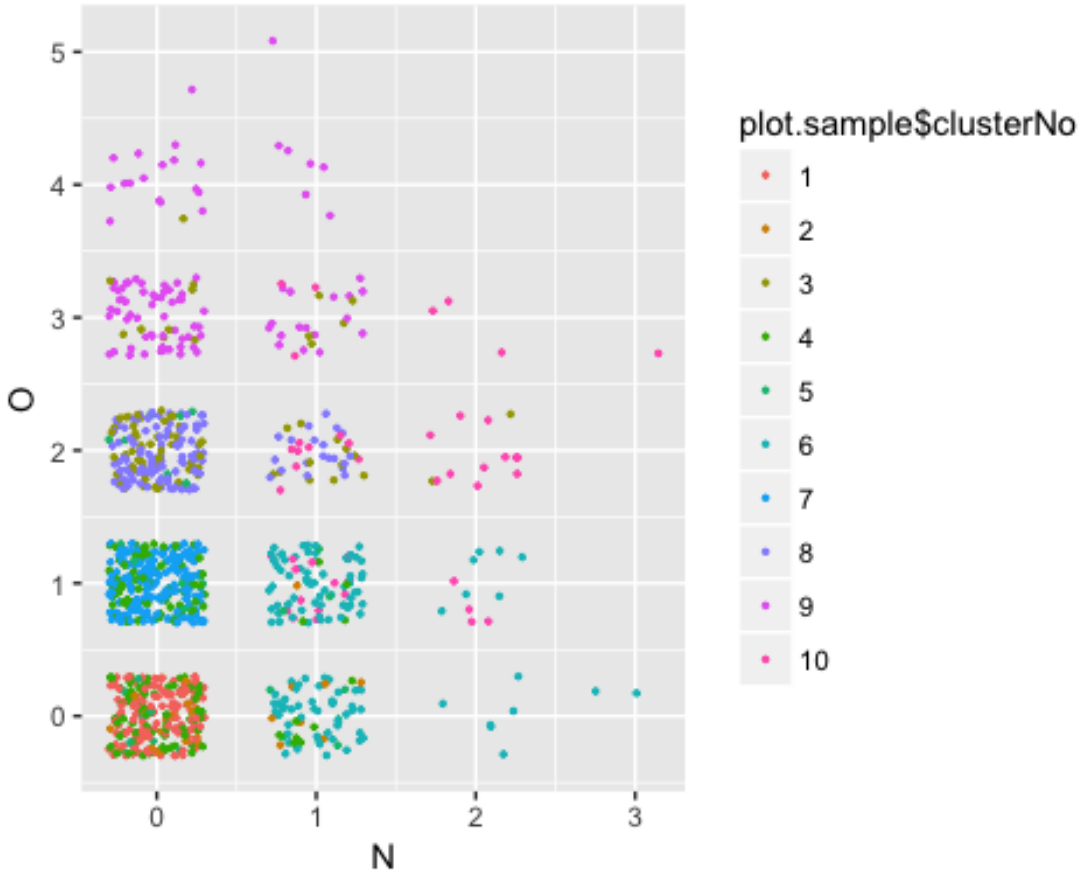
```
## 8  0.16260163 2.00000000 0.18699187 0.15447154 0.0000000 0.10569106
## 9  0.24742268 3.26804124 0.19587629 0.08247423 0.2989691 0.10309278
## 10 1.45454545 1.77272727 0.95454545 0.09090909 0.1590909 0.18181818
##            Os
## 1  0.12096774
## 2  0.18181818
## 3  0.02500000
## 4  0.07692308
## 5  0.11111111
## 6  0.12878788
## 7  0.15428571
## 8  0.13821138
## 9  0.07216495
## 10 0.11363636

plot.sample$clusterNo <- as.factor(kmeans.clusters$cluster)

ggplot(plot.sample, aes(N, P, color = plot.sample$clusterNo)) +
  geom_point(size = 0.5, position = position_jitter(height = 0.3, width =
0.3))
```



Of course, I wouldn't know how many clusters to initialize on. But I had hope a hint of some structure in the data would be visible here. This is obviously not the case.

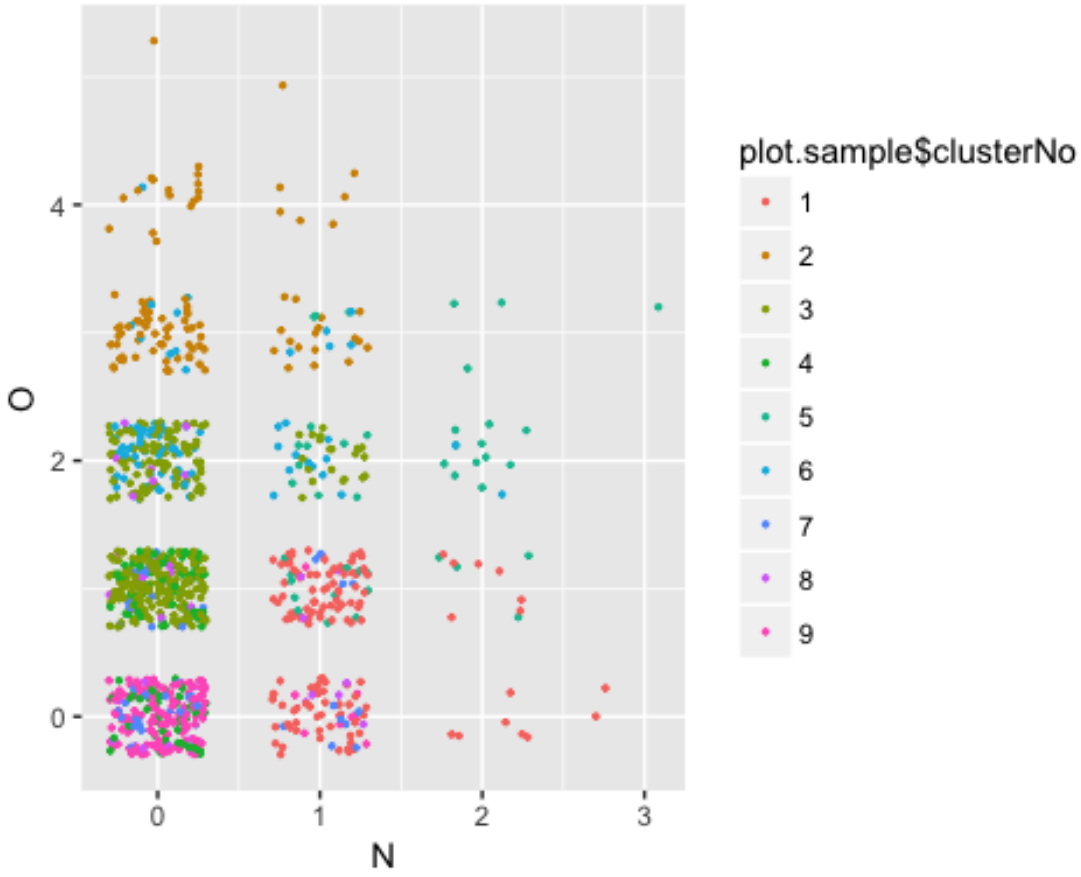
Funny enough, when plotting two other dimensions, some data structure does actually appear:

```
ggplot(plot.sample, aes(N, O, color = plot.sample$clusterNo)) +
  geom_point(size = 0.5, position = position_jitter(height = 0.3, width =
0.3))
```

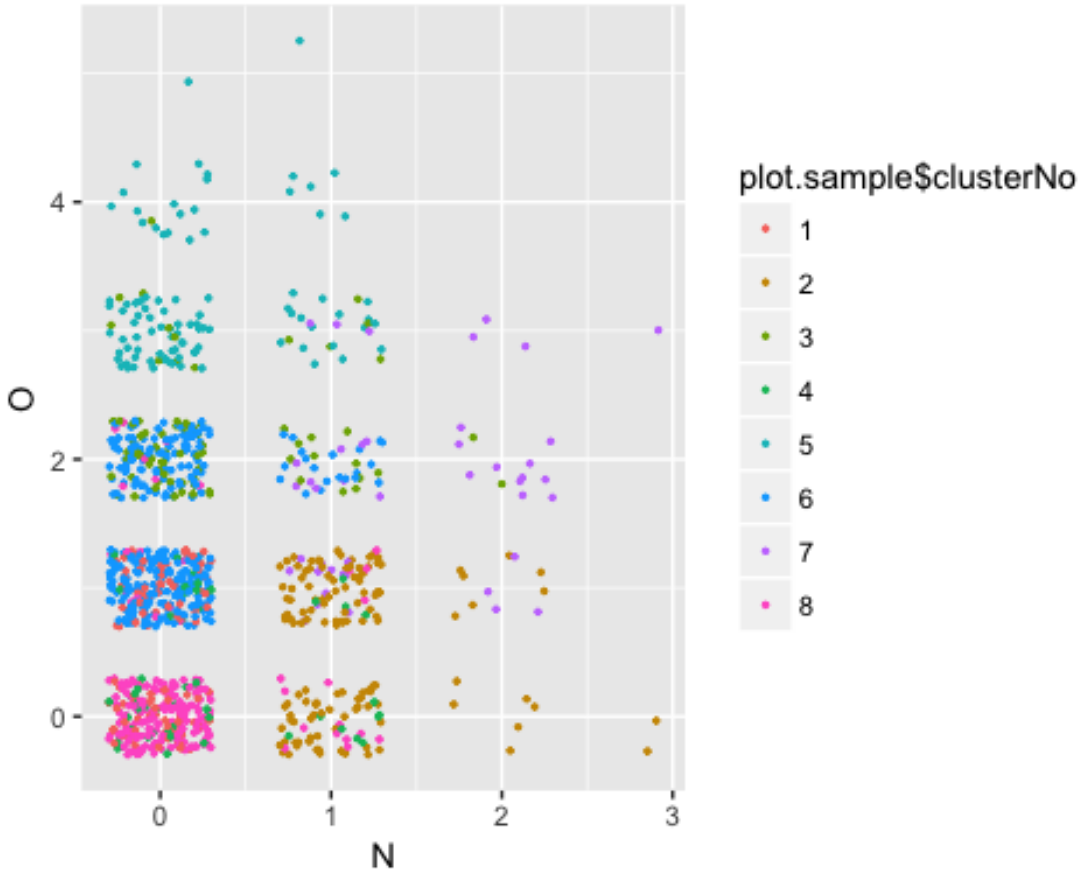


Maybe there are too many clusters?

```
kmeans.clusters <- kmeans(plot.sample, centers = 9, nstart = 20)
plot.sample$clusterNo <- as.factor(kmeans.clusters$cluster)

ggplot(plot.sample, aes(N, O, color = plot.sample$clusterNo)) +
  geom_point(size = 0.5, position = position_jitter(height = 0.3, width =
0.3))
```




```
kmeans.clusters <- kmeans(plot.sample, centers = 8, nstart = 20)
plot.sample$clusterNo <- as.factor(kmeans.clusters$cluster)

ggplot(plot.sample, aes(N, O, color = plot.sample$clusterNo)) +
  geom_point(size = 0.5, position = position_jitter(height = 0.3, width =
0.3))
```


O
4
2
0
0
1
2
3
N
plot.sample$clusterNo
1
2
3
4
5
6
7
8