Clustering

Tony Ni, Antonella Basso, Jose Lopez

6/21/2020

Libraries

Reading in Data

```
setwd("~/harvard-summer-biostats")
df <- read_csv("data/wide_illinois.csv") #read in data
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   well_id = col_character(),
##   site = col_character(),
##   disposal_area = col_character(),
##   type = col_character(),
##   gradient = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
df1 <- na.omit(df) #get rid of na's

df1 <- df1[-69, ]
```

Hierarchical Methods

If we want to look for cereal groups via hierarchical clustering, we need to construct a distance matrix. Distances are constructed with the *dist* function, and we need to choose whether we compute them on scaled or unscaled variables (standardize or not).

```
df.dist <- dist(df1[, -c(1:5)])
```

Now we look at how hierarchical clustering is applied. The relevant function is *hclust*. We can look at the dendrogram, also.

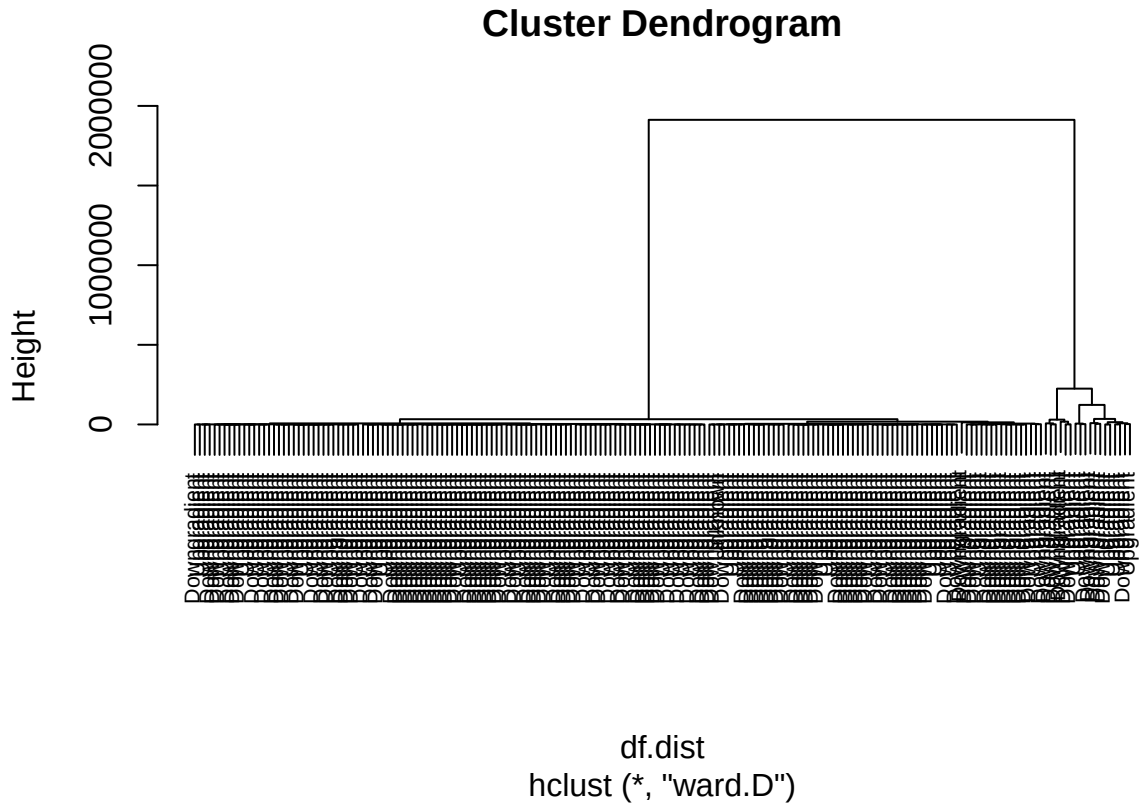
The options for hclust in terms of linkages are provided in the help under options for method. The following options are listed: “ward.D”, “ward.D2”, “single”, “complete”, “average”, “mcquitty”, “median” or “centroid”.

In order to obtain cluster labels, we need to *cut* our dendrograms.

To learn more details about the clusters we found:

Arsenic, Lithium, Boron are some widely found contaminants whose concentrations are often above the thresholds considered to be safe.

```
hcward <- hclust(df.dist, method = "ward.D")
plot(hcward, labels = df1$gradient, cex = 0.7) #cex adjusts size of label
```




```
wardSol <- (cutree(hcward, k = 2)) #cluster labels are numeric, k= # clusters

knitr::kable(table(summary(as.factor(wardSol)))) #as factor to get table
```

Var1	Freq
18	1
172	1

```
favstats(Boron ~ wardSol, data = df1) #can choose any variable
```

```
##   wardSol      min          Q1      median         Q3   max       mean
## 1       1  0.01000  0.06881944   0.3486667   1.787194  60.9   1.879893
## 2       2 31.28651 86.09861111 118.7777778 144.569444 195.0 114.489150
##          sd   n missing
## 1  5.221868 172       0
```

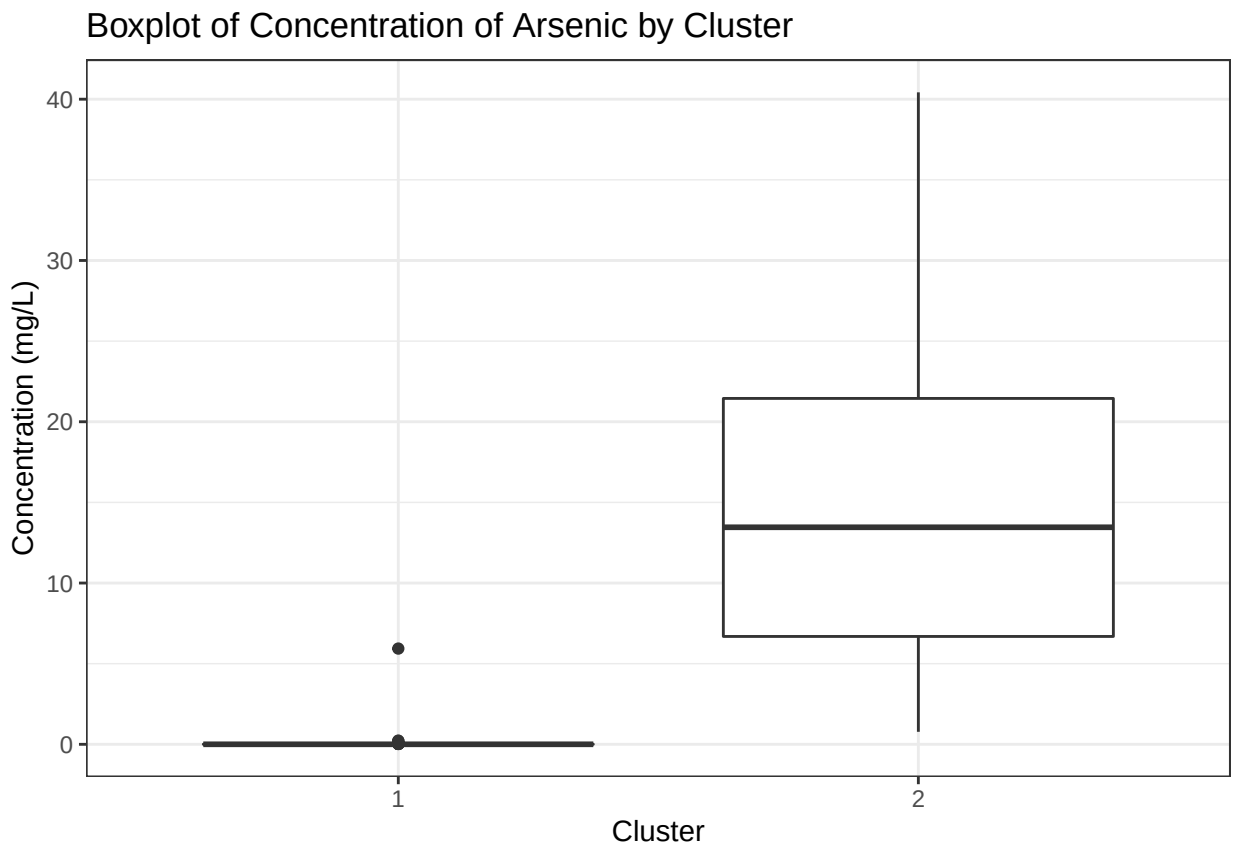
```
## 2 41.134146  18       0
```

```
ggplot(df1, aes(x = as.factor(wardSol), y = Boron)) +
  geom_boxplot() +
  theme_bw() +
  ggtitle("Boxplot of Concentration of Boron by Cluster") +
  xlab("Cluster") +
  ylab("Concentration (mg/L)")
```


```
favstats(Arsenic ~ wardSol, data = df1) #can choose any variable
```

```
##   wardSol       min       Q1     median          Q3    max        mean
## 1       1 0.0005250 0.001015  0.0021125  0.00843125  5.936  0.04588403
## 2       2 0.7757625 6.687500 13.4565000 21.44687500 40.425 15.48252257
##           sd   n missing
## 1  0.4526302 172       0
## 2 13.1701676  18       0
```

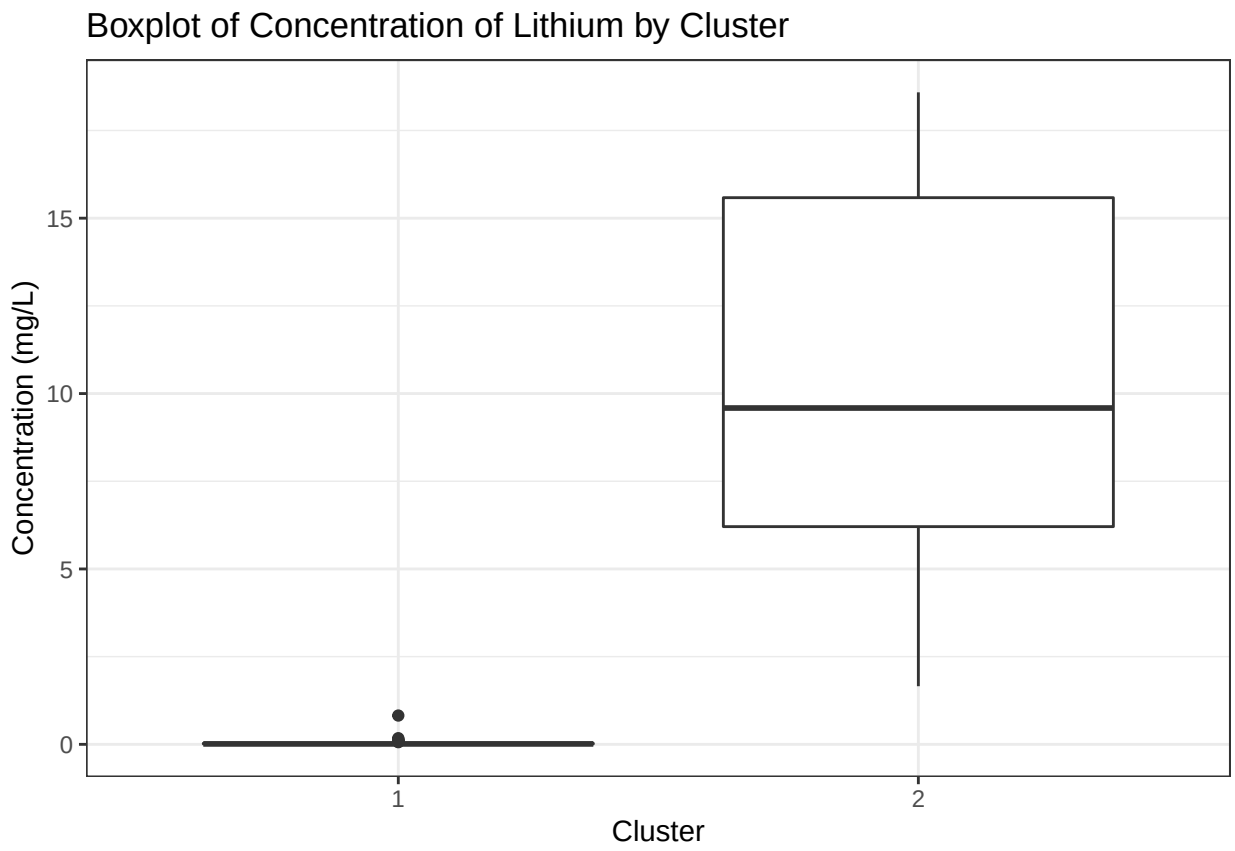
```
ggplot(df1, aes(x = as.factor(wardSol), y = Arsenic)) +
  geom_boxplot() +
  theme_bw() +
  ggtitle("Boxplot of Concentration of Arsenic by Cluster") +
  xlab("Cluster") +
  ylab("Concentration (mg/L)")
```




```
favstats(Lithium ~ wardSol, data = df1) #can choose any variable
```

```
##   wardSol      min       Q1     median          Q3        max      mean
## 1       1 0.000875 0.010000 0.01185938  0.03080625  0.8207778 0.0285163
## 2       2 1.657163 6.206403 9.58750000 15.58437500 18.5875000 9.9628625
##           sd   n missing
## 1 0.06644773 172       0
## 2 5.56052280  18       0
```

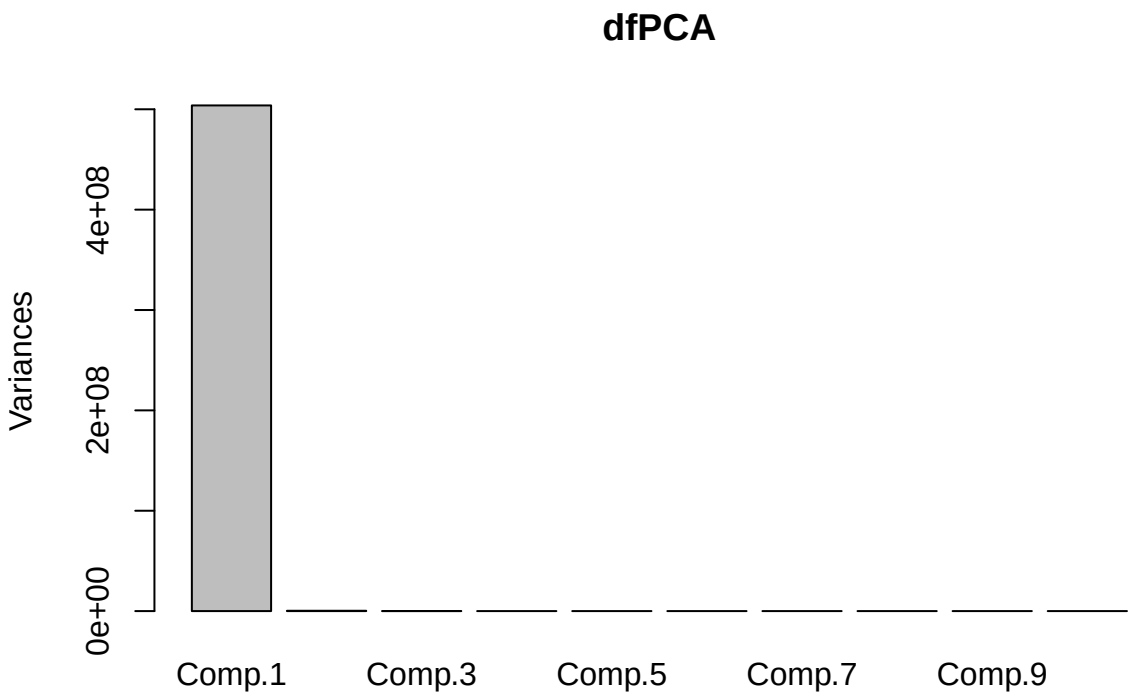
```
ggplot(df1, aes(x = as.factor(wardSol), y = Lithium)) +
  geom_boxplot() +
  theme_bw() +
  ggtitle("Boxplot of Concentration of Lithium by Cluster") +
  xlab("Cluster") +
  ylab("Concentration (mg/L)")
```



Our cluster sizes are extremely uneven... Our first clsuter has 172 wells and the second has 19.

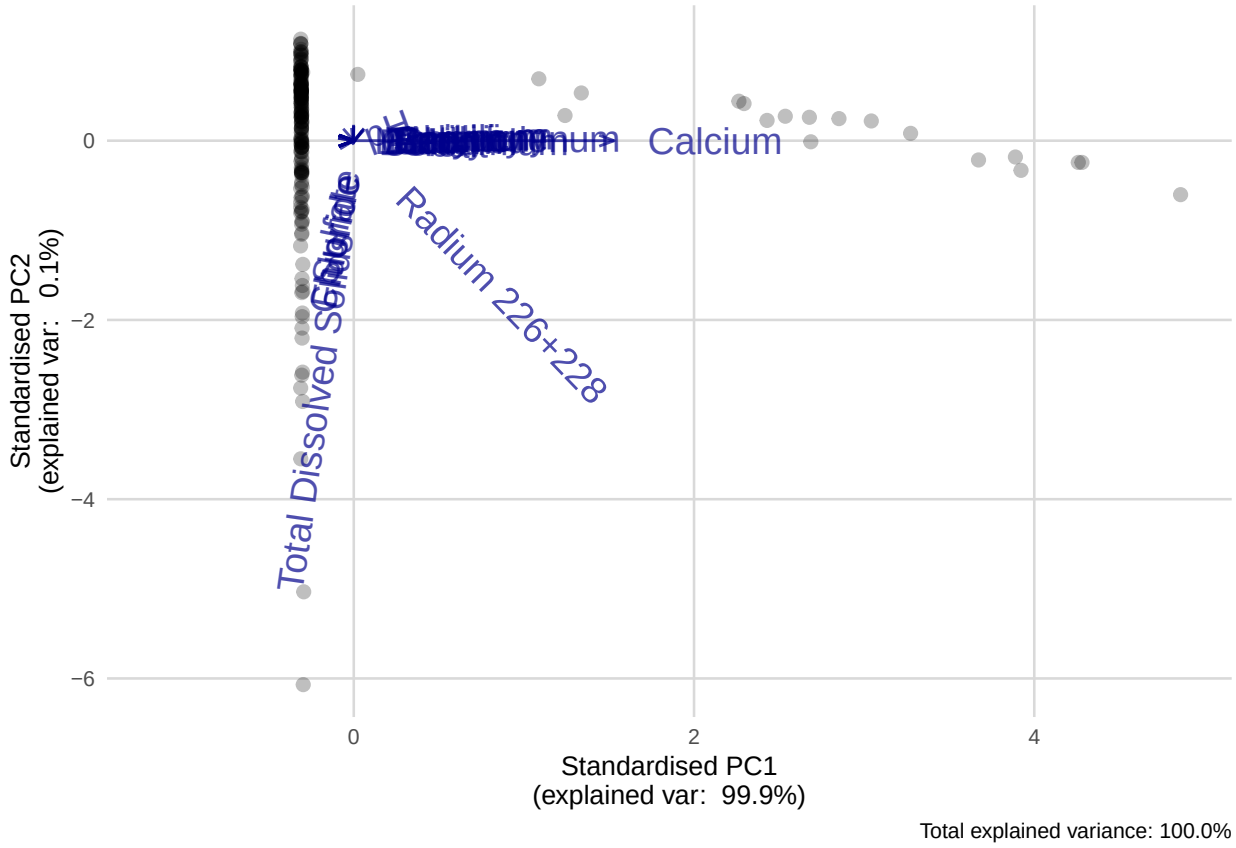
We can view the solution in the PC space (say 2-D) to see how well-separated the clusters are in that space. Because we used an unstandardized distance, we will run the PCA on the covariance matrix.

```
dfPCA <- princomp(df1[, -c(1:5)], cor = FALSE)
plot(dfPCA)
```



From the scree plot we generated above, we can see that the first PC captures essentially all the variation within our dataset.

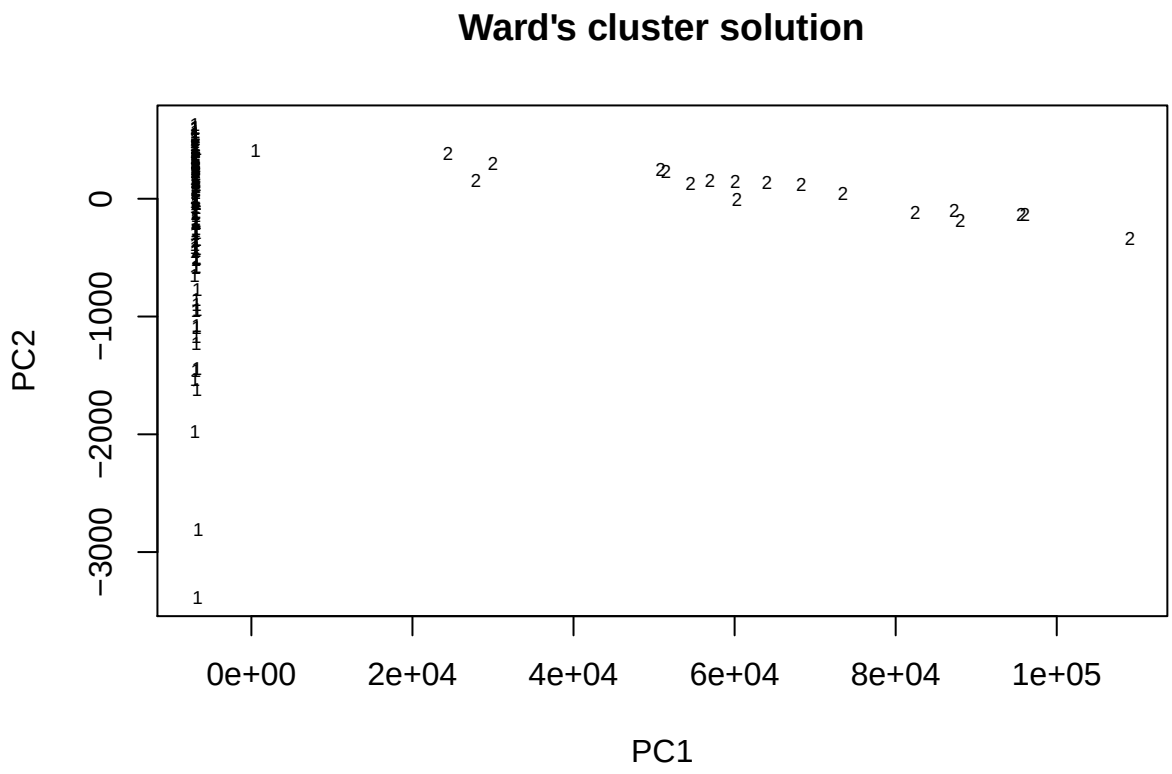
```
AMR::ggplot_pca(dfPCA, base_textsize = 10, arrows_textsize = 5, arrows_alpha = 0.7)
```



Like we expected from our scree plot, all of the variance is explained in the first principal component. Since PC1 was heavily dominated by almost every single variable in the dataset - it could potentially indicate that it explains some sort of weighted average.

When two vectors are close, forming a small angle, the two variables they represent are positively correlated – we can see that every variable seems to be positively correlated with one another, except for radium 226+228 and Total Dissolved Solids. The right angle that the Total Dissolved Solids vector has with all the other contaminants gives us reason to believe that it has no correlation to any of the contaminants apart from Raidum 226+228.

```
plot(dfPCA$scores[, 1:2], type = "n", xlab = "PC1", ylab = "PC2", main = "Ward's cluster solution") #bl
text(dfPCA$scores[, 1:2], labels = wardSol, cex = 0.6) #add the text
```



A plot of our Ward's cluster solution shows that these wells do seem to be well separated from one another.

We may want to go into investigation to see what sort of traits/attributes are shared by the wells in each cluster and seeing if we can find meaning in them.

K-means Methods

For k-means, we don't need to compute the distance matrix ourselves. We feed the function the data set to operate on:

```
Ksol1 <- kmeans(scale(df1[, -c(1:5)]), centers = 2) #centers is the # of clusters desired
list(Ksol1) #so you can see what it gives you
```

```
## [[1]]
## K-means clustering with 2 clusters of sizes 15, 175
##
## Cluster means:
##     Antimony    Arsenic     Barium  Beryllium      Boron    Cadmium    Calcium
## 1  3.3166216  2.5185136  2.7542737  3.3165006  3.2073866  3.3164884  3.2539443
## 2 -0.2842818 -0.2158726 -0.2360806 -0.2842715 -0.2749189 -0.2842704 -0.2789095
##      Chloride   Chromium     Cobalt    Fluoride       Lead    Lithium
## 1 -0.39372731  2.0374723  2.4266397 -0.13992310  2.3798126  3.0723177
## 2  0.03374806 -0.1746405 -0.2079977  0.01199341 -0.2039839 -0.2633415
##      Mercury Molybdenum          pH Radium 226+228   Selenium     Sulfate
## 1  3.3164896  2.5553332  0.58147104     0.32016443  3.2236298 -0.54045694
```

```
## 2 -0.2842705 -0.2190286 -0.04984037    -0.02744267 -0.2763111  0.04632488
##     Thallium Total Dissolved Solids
## 1  3.3151785            -0.47664690
## 2 -0.2841582             0.04085545
##
## Clustering vector:
##   [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [38] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 1 2 2 1 1 2 2 2 2 2 2
##  [75] 2 1 2 2 2 2 2 1 1 2 1 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [112] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [149] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 2
## [186] 1 2 2 2 2
##
## Within cluster sum of squares by cluster:
## [1]  576.1418 1227.0094
##  (between_SS / total_SS =  54.6 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

The list option provides us with lots of information. We can pull out the cluster means as:

```
Ksol1$centers
```

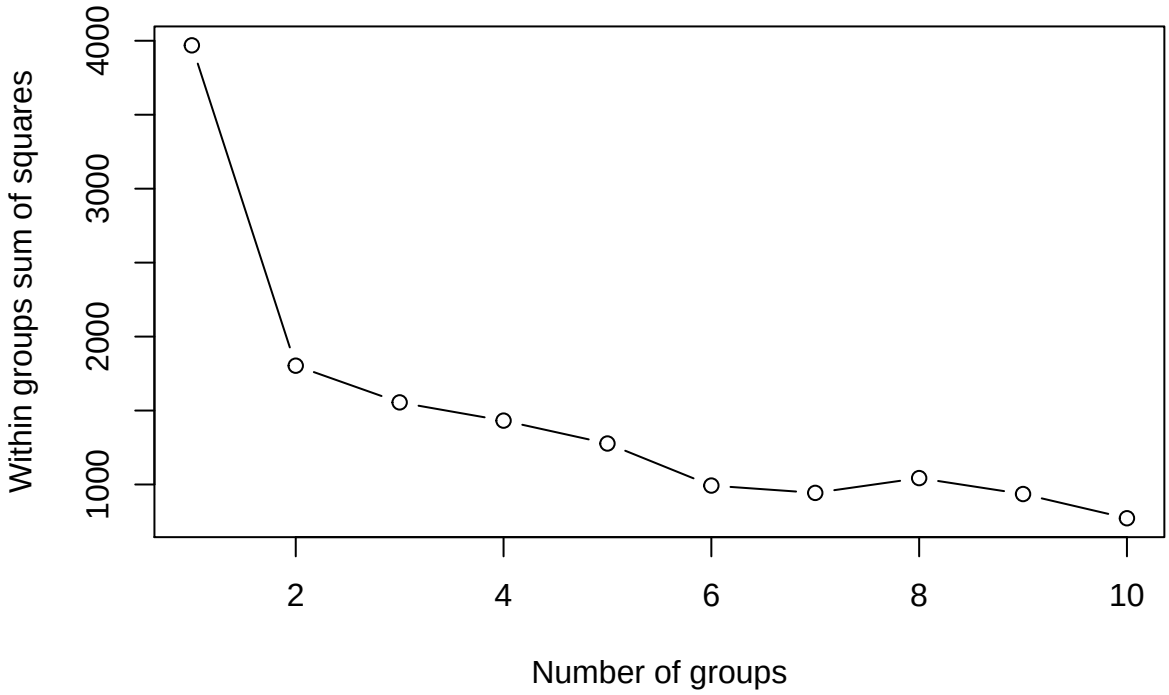
```
##     Antimony    Arsenic     Barium  Beryllium      Boron    Cadmium    Calcium
## 1  3.3166216  2.5185136  2.7542737  3.3165006  3.2073866  3.3164884  3.2539443
## 2 -0.2842818 -0.2158726 -0.2360806 -0.2842715 -0.2749189 -0.2842704 -0.2789095
##      Chloride   Chromium     Cobalt    Fluoride       Lead    Lithium
## 1 -0.39372731  2.0374723  2.4266397 -0.13992310  2.3798126  3.0723177
## 2  0.03374806 -0.1746405 -0.2079977  0.01199341 -0.2039839 -0.2633415
##      Mercury Molybdenum          pH Radium 226+228   Selenium     Sulfate
## 1  3.3164896  2.5553332  0.58147104     0.32016443  3.2236298 -0.54045694
## 2 -0.2842705 -0.2190286 -0.04984037    -0.02744267 -0.2763111  0.04632488
##     Thallium Total Dissolved Solids
## 1  3.3151785            -0.47664690
## 2 -0.2841582             0.04085545
```

In order to determine if we have chosen a "good" value of the number of clusters, we can look at the within cluster sum of squares for this solution and a few other options for k, the number of clusters. This runs the solution from 1 to 10 clusters and pulls the within group sum of squares from each.

```
n <- nrow(df1) #number of observations

wss <- rep(0, 10) #creates 10 copies of 0 to create an empty vector
for(i in 1:10){
  wss[i] <- sum(kmeans(scale(df1[, -c(1:5)]), centers = i)$withinss)
}

plot(1:10, wss, type = "b", xlab = "Number of groups", ylab = "Within groups sum of squares")
```



We look for elbows in the plot - here there are elbows at 2 and 6 (ish?), maybe these values will be good to use?

With two clusters, we should see if there is any relationship with sites...

```
tally(Ksol1$cluster ~ type, data = df1, format = "count")
```

```
##              type
## Ksol1$cluster   L   M  SI
##             1  15   0   0
##             2  32   5 138
```

There does seem to be something of interest here... The first cluster has a varied mix between all types with the majority in SI, while wells in cluster 2 seem to only consist of L.

We can compare clustering solutions with similar tables. How do the K-means and Ward's solutions overlap?

```
tally(Ksol1$cluster ~ wardSol, data = df1, format = "count")
```

```
##              wardSol
## Ksol1$cluster   1   2
##             1   0  15
##             2 172   3
```

They seem to match up fairly well!

Can we try some sort of clustering algorithm where we don't have the specifiy the number of clusters (it automatically detects it for us? so that it might be able to differentiate between different severity/intensity levels of contamination)

Alternative Approaches

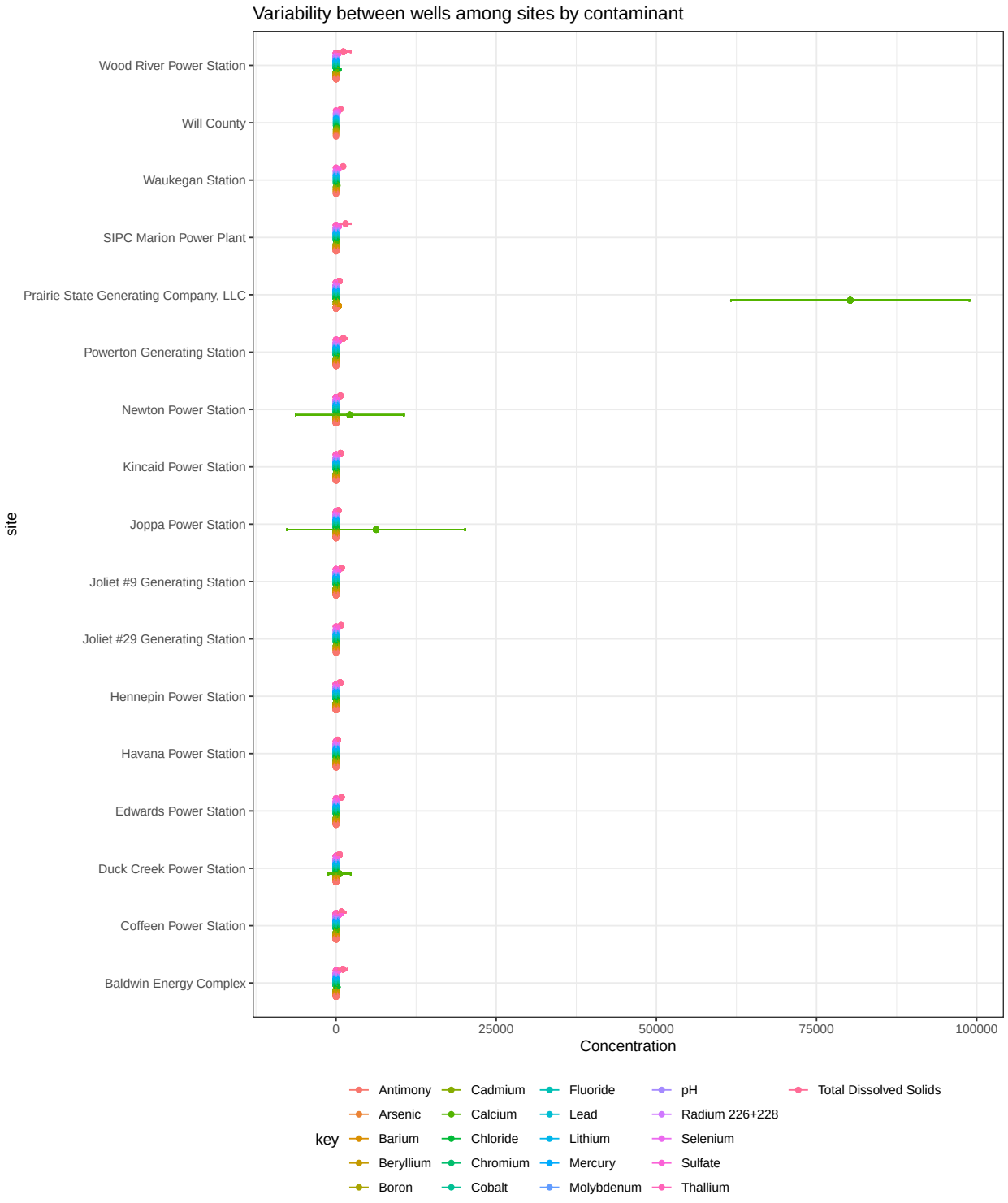
"If you haven't already, you can try some alternative filtering approaches: there might be a lot of contaminants that don't vary that much among the wells, so one thing you could do is find the contaminants with the highest between-well variability. For each chemical you can compute the variance across wells within a site, and take only the some number of contaminants (10? 20? Or 30?) with the highest variance, and then use those chemicals to do the PCA/k-means/hierarchical clustering."

```
df2 <- df1 %>%
  gather(key, value, c(6:26)) %>%
  group_by(site, key) %>%
  mutate(m = mean(value), sd = sd(value)) %>%
  arrange(desc(sd)) %>%
  print
```

```
## # A tibble: 3,990 x 9
## # Groups:   site, key [357]
##    well_id site        disposal_area   type  gradient key    value      m     sd
##    <chr>   <chr>       <chr>           <chr> <chr>    <chr>  <dbl>  <dbl>  <dbl>
##  1 G03D    Prairie St~ Near Field Lan~ L     Upgradi~ Calc~ 5.85e4 80252. 18610.
##  2 G04D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 1.03e5 80252. 18610.
##  3 G05D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 7.53e4 80252. 18610.
##  4 G07D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 9.43e4 80252. 18610.
##  5 G08D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 7.10e4 80252. 18610.
##  6 G10D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 1.03e5 80252. 18610.
##  7 G13D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 9.51e4 80252. 18610.
##  8 G15D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 1.16e5 80252. 18610.
##  9 G17D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 8.94e4 80252. 18610.
## 10 G19D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 6.16e4 80252. 18610.
## # ... with 3,980 more rows
```

```
df2 %>%
  ggplot(.) +
  theme_bw() +
  aes(x = site, y = m, ymin = m - sd, ymax = m + sd, color = key) +
  geom_point(position = position_dodge(width = 0.5)) +
  geom_errorbar(position = position_dodge(width = 0.5)) +
  ylab("Concentration") +
  ggtitle("Variability between wells among sites by contaminant") +
  coord_flip() +
  theme(legend.position="bottom")
```

```
## Warning: Removed 21 rows containing missing values (geom_errorbar).
```

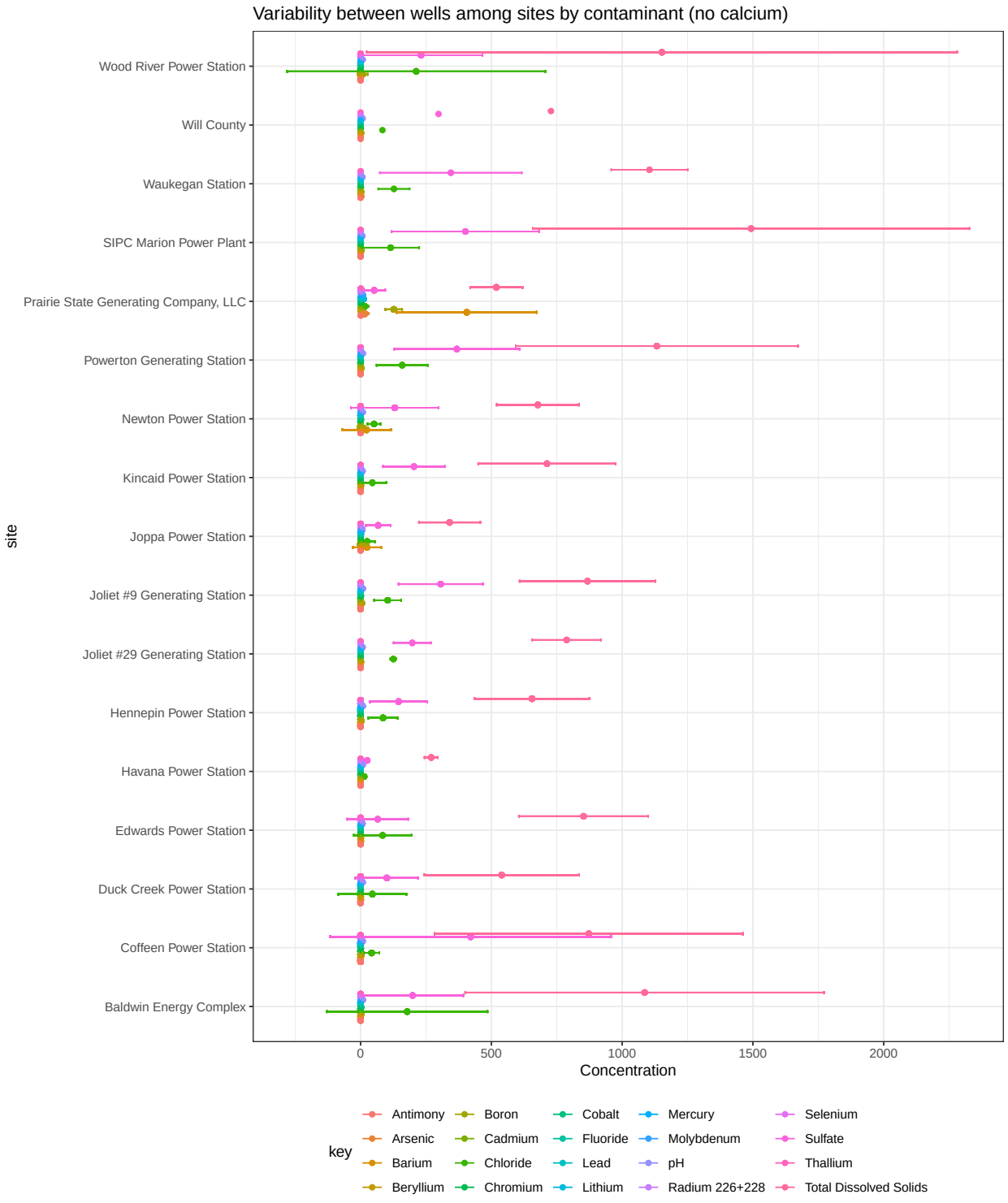


Calcium dominates the entire graph... Remove it...

```
df2 %>%
  filter(key != "Calcium") %>%
  ggplot(.) +
  theme_bw() +
```

```
  aes(x = site, y = m, ymin = m - sd, ymax = m + sd, color = key) +
  geom_point(position = position_dodge(width = 0.5)) +
  geom_errorbar(position = position_dodge(width = 0.5)) +
  ylab("Concentration") +
  ggtitle("Variability between wells among sites by contaminant (no calcium)") +
  coord_flip() +
  theme(legend.position="bottom")
```

```
## Warning: Removed 20 rows containing missing values (geom_errorbar).
```


Variability between wells among sites by contaminant (no calcium)
site
Wood River Power Station
Will County
Waukegan Station
SIPC Marion Power Plant
Prairie State Generating Company, LLC
Powerton Generating Station
Newton Power Station
Kincaid Power Station
Joppa Power Station
Joliet #9 Generating Station
Joliet #29 Generating Station
Hennepin Power Station
Havana Power Station
Edwards Power Station
Duck Creek Power Station
Coffeen Power Station
Baldwin Energy Complex
0
500
1000
1500
2000
Concentration
key
Antimony
Arsenic
Barium
Beryllium
Boron
Cadmium
Chloride
Chromium
Cobalt
Fluoride
Lead
Lithium
Mercury
Molybdenum
pH
Radium 226+228
Selenium
Sulfate
Thallium
Total Dissolved Solids