Clustering

Tony Ni, Antonella Basso, Jose Lopez

6/21/2020

Libraries

Reading in Data

```r
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.
```

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

df1 <- df1[-69, ] %>%
  filter(gradient == "Upgradient")
```

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).

```r
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.

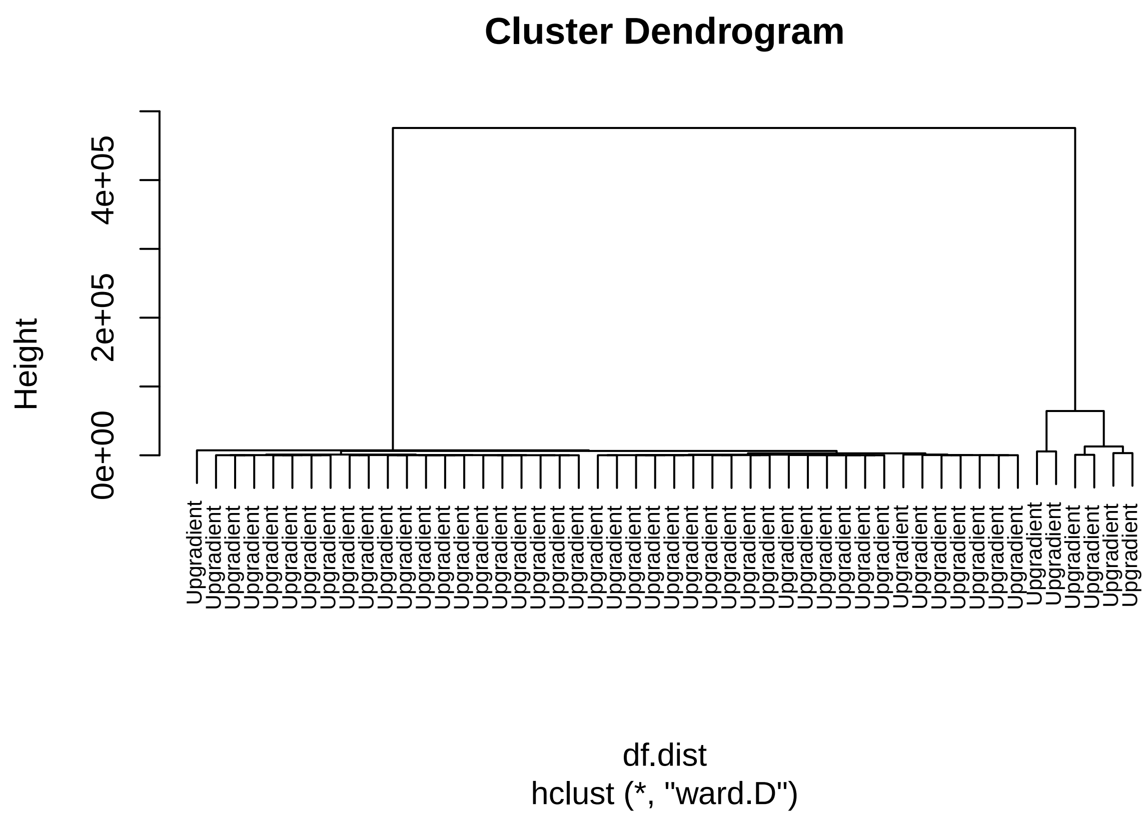
The options for hclust in terms of linkages are provided in the help under options for method. The following options are listed: “ward.D”, “ward.D2”, “single”, “complete”, “average”, “mcquitty”, “median” or “centroid”.

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
```



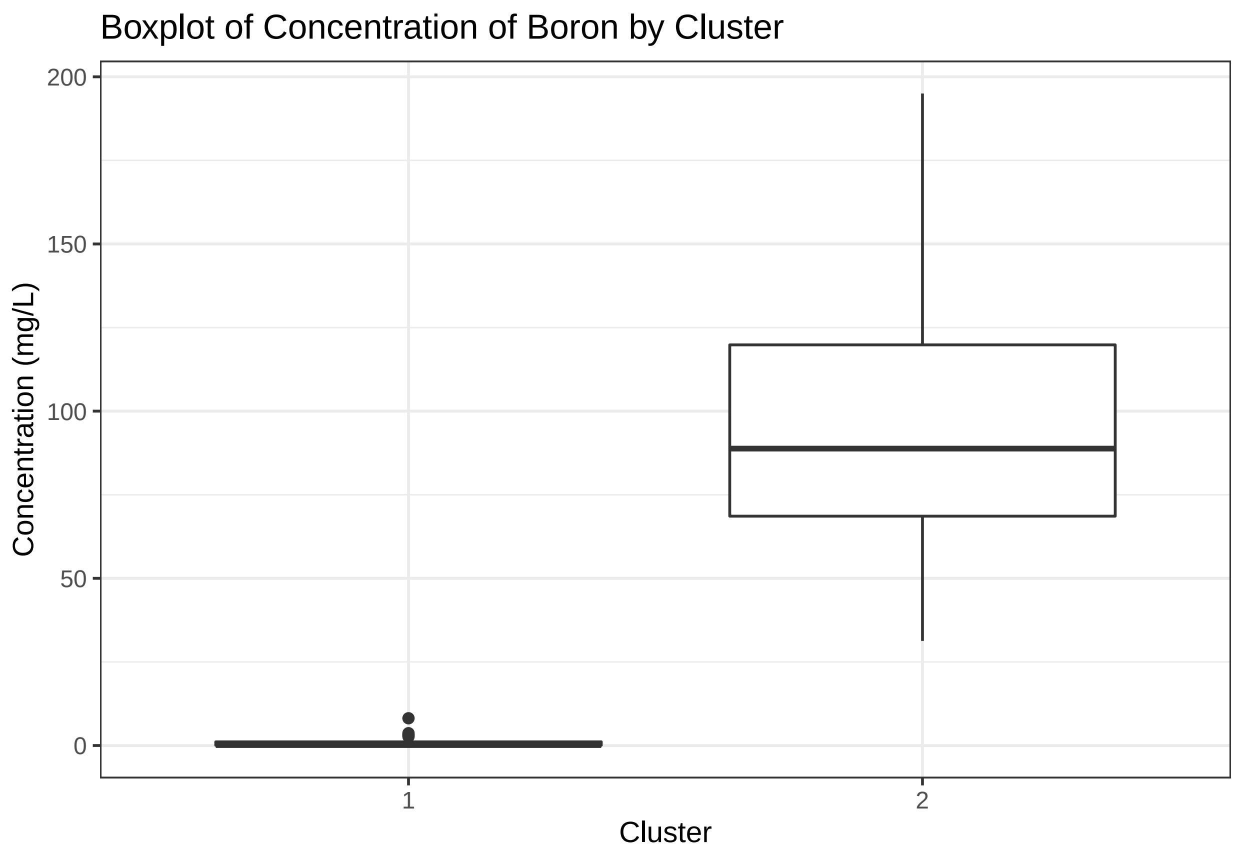
```
wardSol <- (cutree(hcward, k = 2)) #cluster labels are numeric, k= # clusters

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

```
##  1  2
## 44  6
```

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

```
##   wardSol      min          Q1     median         Q3    max       mean
## 1       1  0.01100  0.06387222  0.1220556   1.097083   8.15  0.7916992
## 2       2 31.28651 68.59127639 88.8000000 119.827778 195.00 99.3186528
##          sd  n missing
## 1  1.476491 44       0
## 2 56.791882  6       0
```

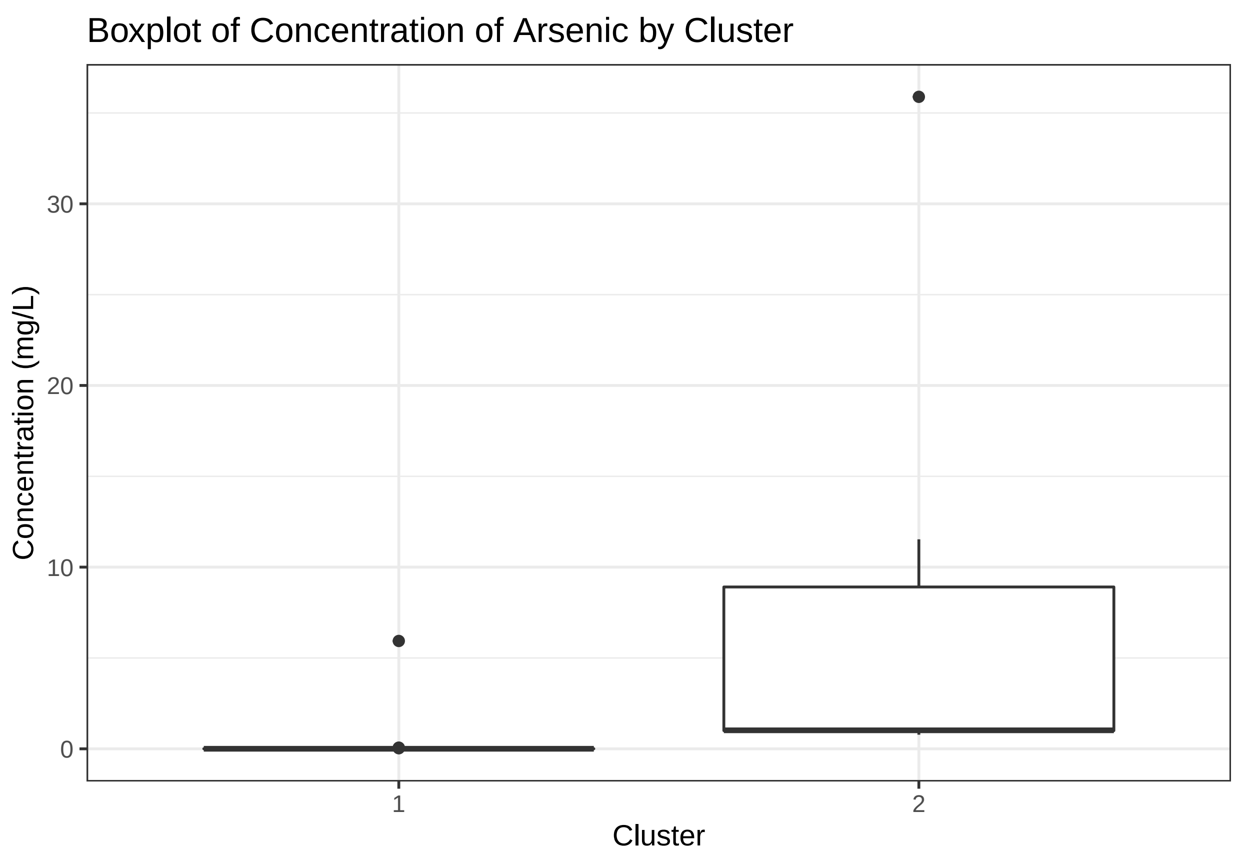
```r
ggplot(df1, aes(x = as.factor(wardSol), y = Boron)) +
  geom_boxplot() +
  theme_bw() +
  ggtitle("Boxplot of Concentration of Boron by Cluster") +
  xlab("Cluster") +
  ylab("Concentration (mg/L)")
```



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

```
##   wardSol       min          Q1  median       Q3     max      mean         sd
## 1       1 0.0006250 0.001269375 0.00257 0.007795  5.9360 0.1418055  0.8939071
## 2       2 0.7757625 1.000000000 1.01875 8.908188 35.8875 8.5387521 14.0506968
##    n missing
## 1 44       0
## 2  6       0
```

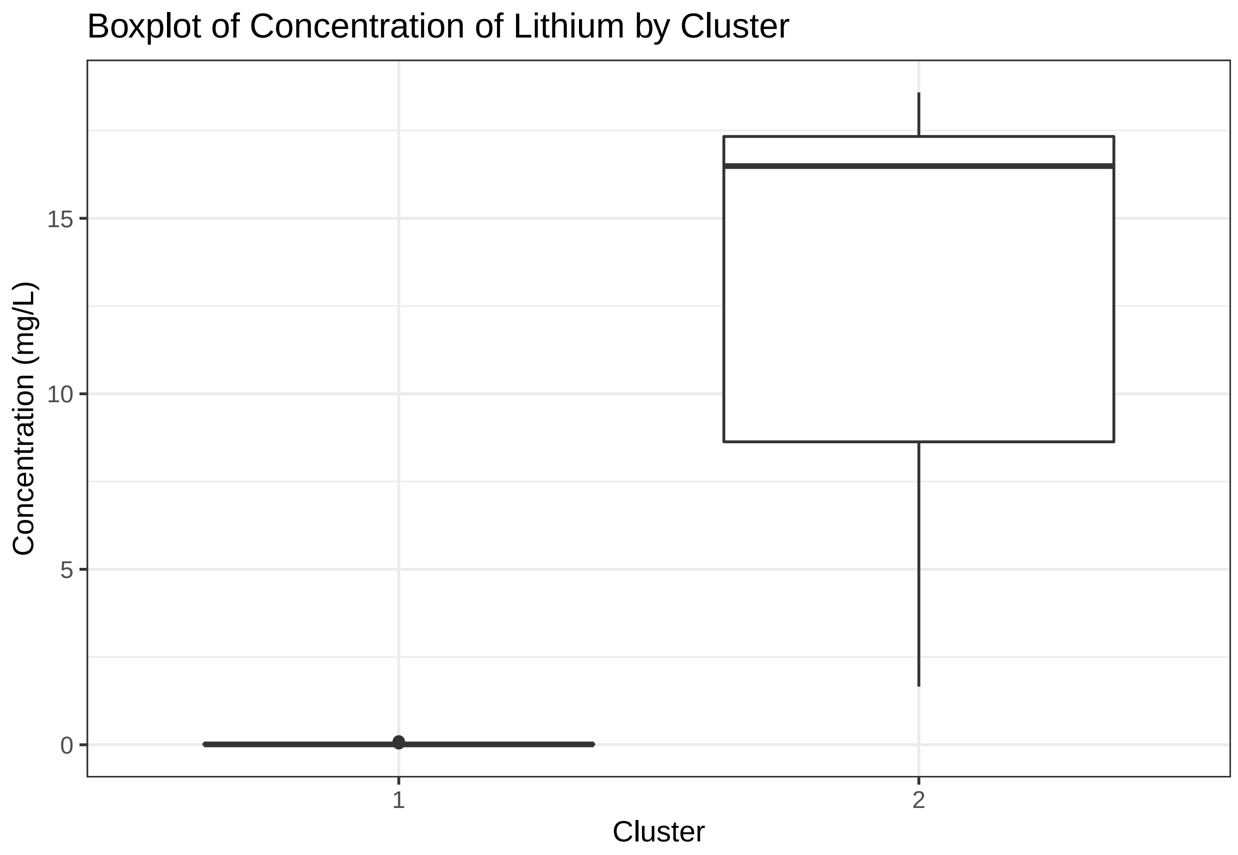
```r
ggplot(df1, aes(x = as.factor(wardSol), y = Arsenic)) +
  geom_boxplot() +
  theme_bw() +
  ggtitle("Boxplot of Concentration of Arsenic by Cluster") +
  xlab("Cluster") +
  ylab("Concentration (mg/L)")
```



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

```
##   wardSol       min       Q1     median         Q3       max        mean
## 1       1 0.0008875 0.010000  0.0104375  0.0213875  0.096425  0.01784942
## 2       2 1.6571625 8.631709 16.4875000 17.3312500 18.587500 12.80754583
##           sd  n missing
## 1 0.01829948 44       0
## 2 7.06435077  6       0
```

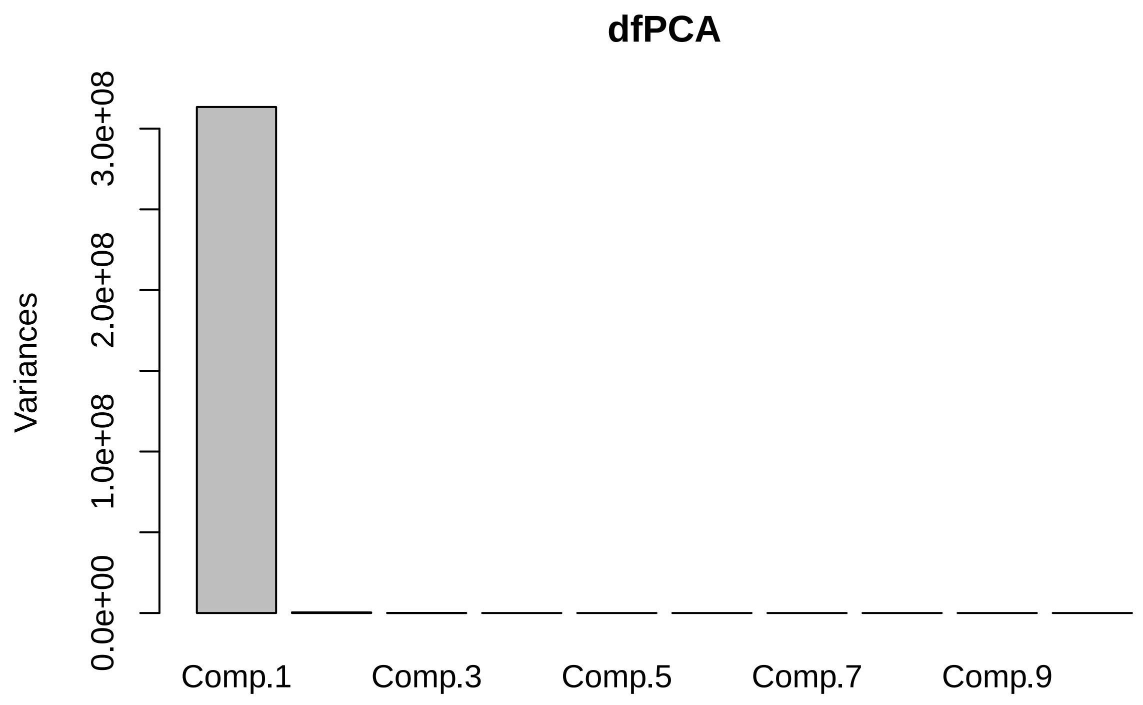
```
ggplot(df1, aes(x = as.factor(wardSol), y = Lithium)) +
  geom_boxplot() +
  theme_bw() +
  ggtitle("Boxplot of Concentration of Lithium by Cluster") +
  xlab("Cluster") +
  ylab("Concentration (mg/L)")
```



Our cluster sizes are extremely uneven… Our first clsuter has 44 wells and the second has 6.

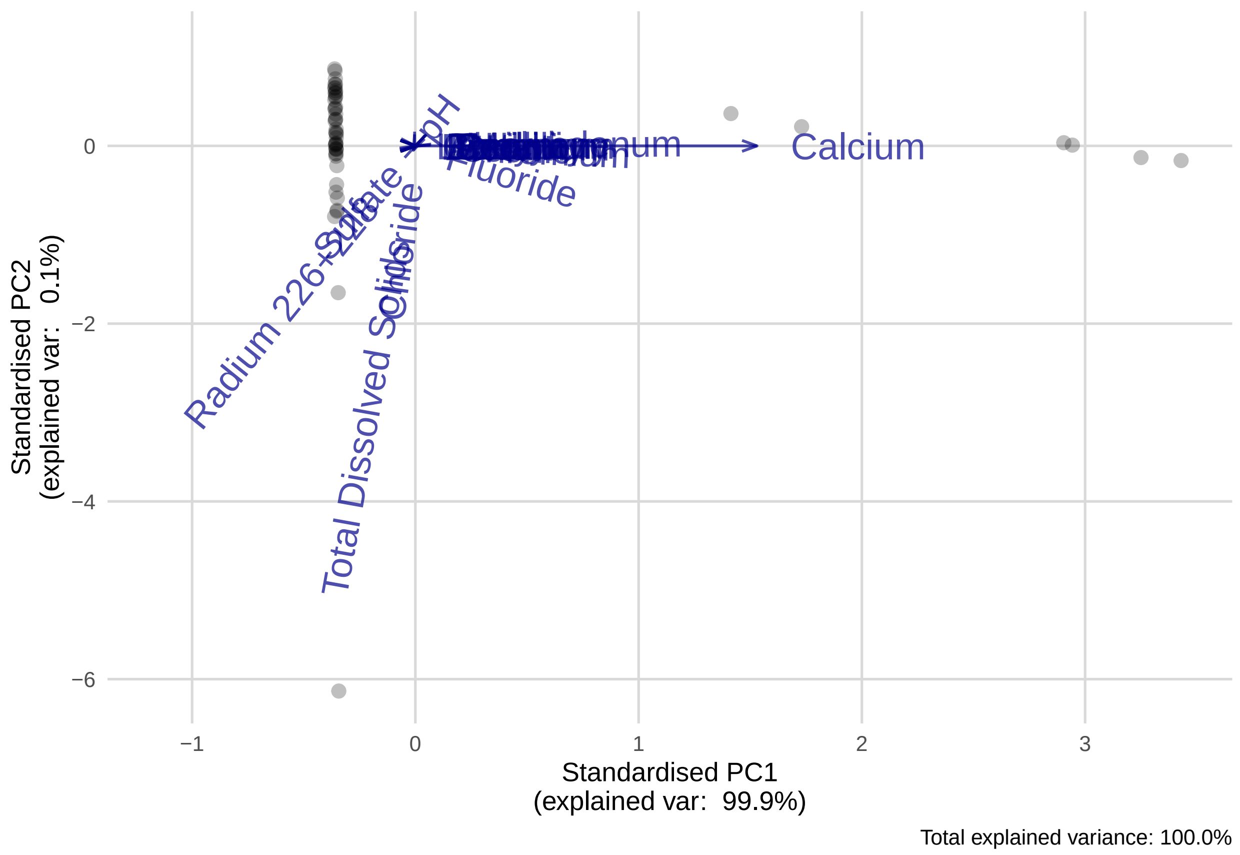
We can view the solution in the PC space (say 2-D) to see how well-separated the clusters are in that space. Because we used an unstandardized distance, we will run the PCA on the covariance matrix.

```r
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.

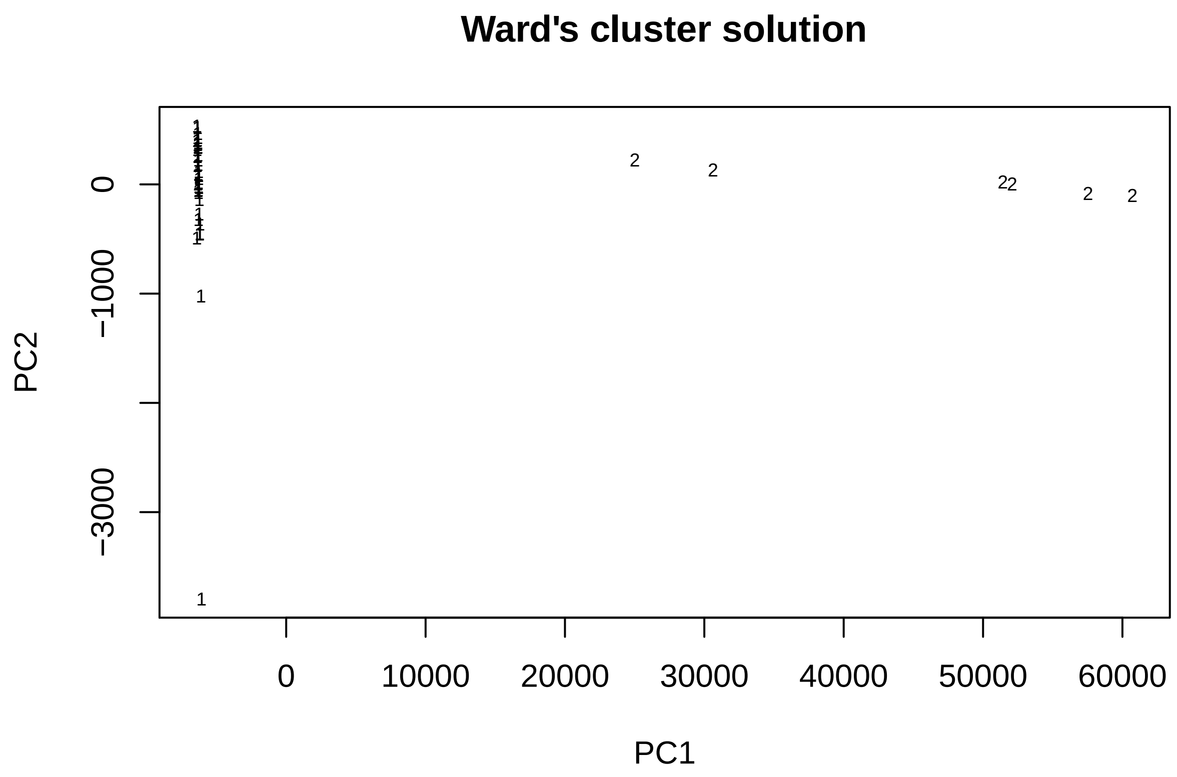
```
AMR::ggplot_pca(dfPCA, base_textsize = 10, arrows_textsize = 5, arrows_alpha = 0.7)
```



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. A right angle between vectors show no correlation.

```r
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.

```
tally(wardSol ~ type, data = df1, format = "count")
```

```
##        type
## wardSol  L  M SI
##       1  6  5 33
##       2  4  0  2
```

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 44, 6
##
## Cluster means:
##    Antimony    Arsenic     Barium  Beryllium      Boron     Cadmium     Calcium
## 1 -0.349952 -0.1889327 -0.3143055 -0.3499968 -0.3186096 -0.3499934 -0.3525854
## 2  2.566315  1.3855068  2.3049073  2.5666429  2.3364705  2.5666181  2.5856265
##      Chloride   Chromium     Cobalt   Fluoride       Lead    Lithium    Mercury
## 1  0.05155071 -0.2801107 -0.3449376 -0.0256245 -0.3479853 -0.3219945 -0.3499975
## 2 -0.37803856  2.0541448  2.5295424  0.1879130  2.5518926  2.3612932  2.5666480
##   Molybdenum          pH Radium 226+228   Selenium    Sulfate   Thallium
## 1 -0.3255934 -0.01368733      0.0753027 -0.3499565  0.1152690 -0.3498814
## 2  2.3876850  0.10037374     -0.5522198  2.5663474 -0.8453058  2.5657972
##   Total Dissolved Solids
## 1             0.06989437
## 2            -0.51255869
##
## Clustering vector:
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [39] 1 1 1 1 1 1 1 2 2 2 1 1
##
## Within cluster sum of squares by cluster:
## [1] 297.9869 125.7456
##  (between_SS / total_SS =  58.8 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

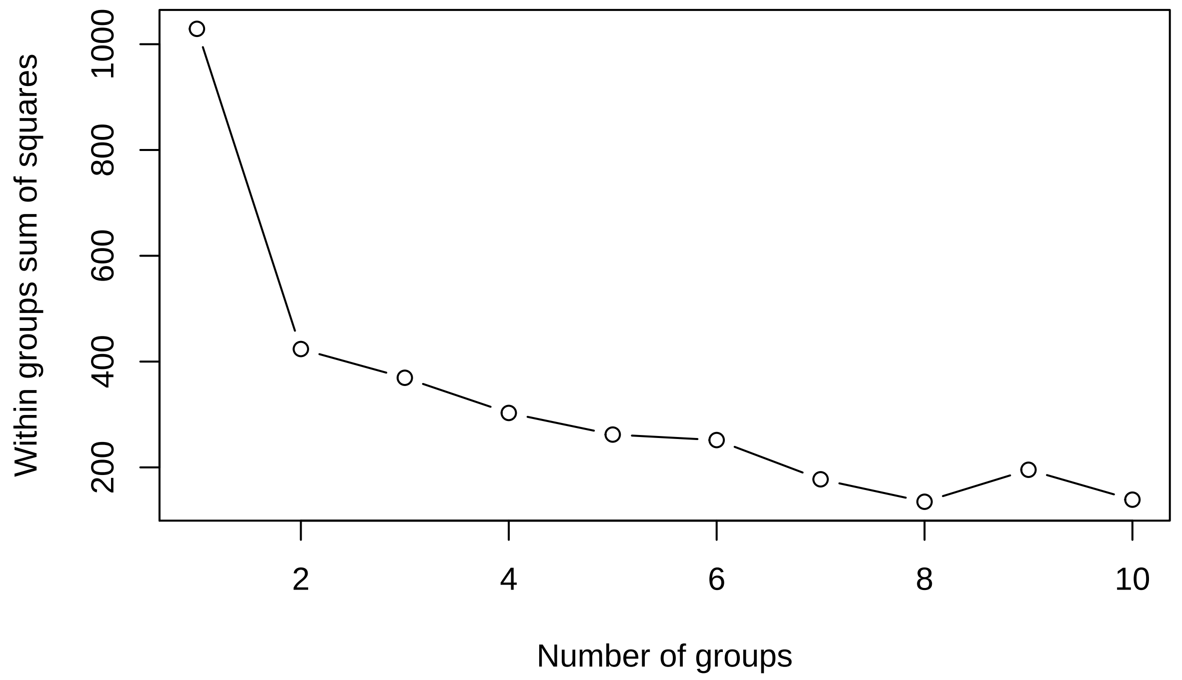
The list option provides us with lots of information. We can pull out the cluster means as:

```
Ksol1$centers
```

```
##    Antimony    Arsenic     Barium  Beryllium      Boron    Cadmium    Calcium
## 1 -0.349952 -0.1889327 -0.3143055 -0.3499968 -0.3186096 -0.3499934 -0.3525854
## 2  2.566315  1.3855068  2.3049073  2.5666429  2.3364705  2.5666181  2.5856265
##      Chloride   Chromium     Cobalt   Fluoride       Lead    Lithium    Mercury
## 1  0.05155071 -0.2801107 -0.3449376 -0.0256245 -0.3479853 -0.3219945 -0.3499975
## 2 -0.37803856  2.0541448  2.5295424  0.1879130  2.5518926  2.3612932  2.5666480
##   Molybdenum          pH Radium 226+228   Selenium    Sulfate   Thallium
## 1 -0.3255934 -0.01368733      0.0753027 -0.3499565  0.1152690 -0.3498814
## 2  2.3876850  0.10037374     -0.5522198  2.5663474 -0.8453058  2.5657972
##   Total Dissolved Solids
## 1             0.06989437
## 2            -0.51255869
```

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?

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 44  0
##             2  0  6
```

They match exactly!

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

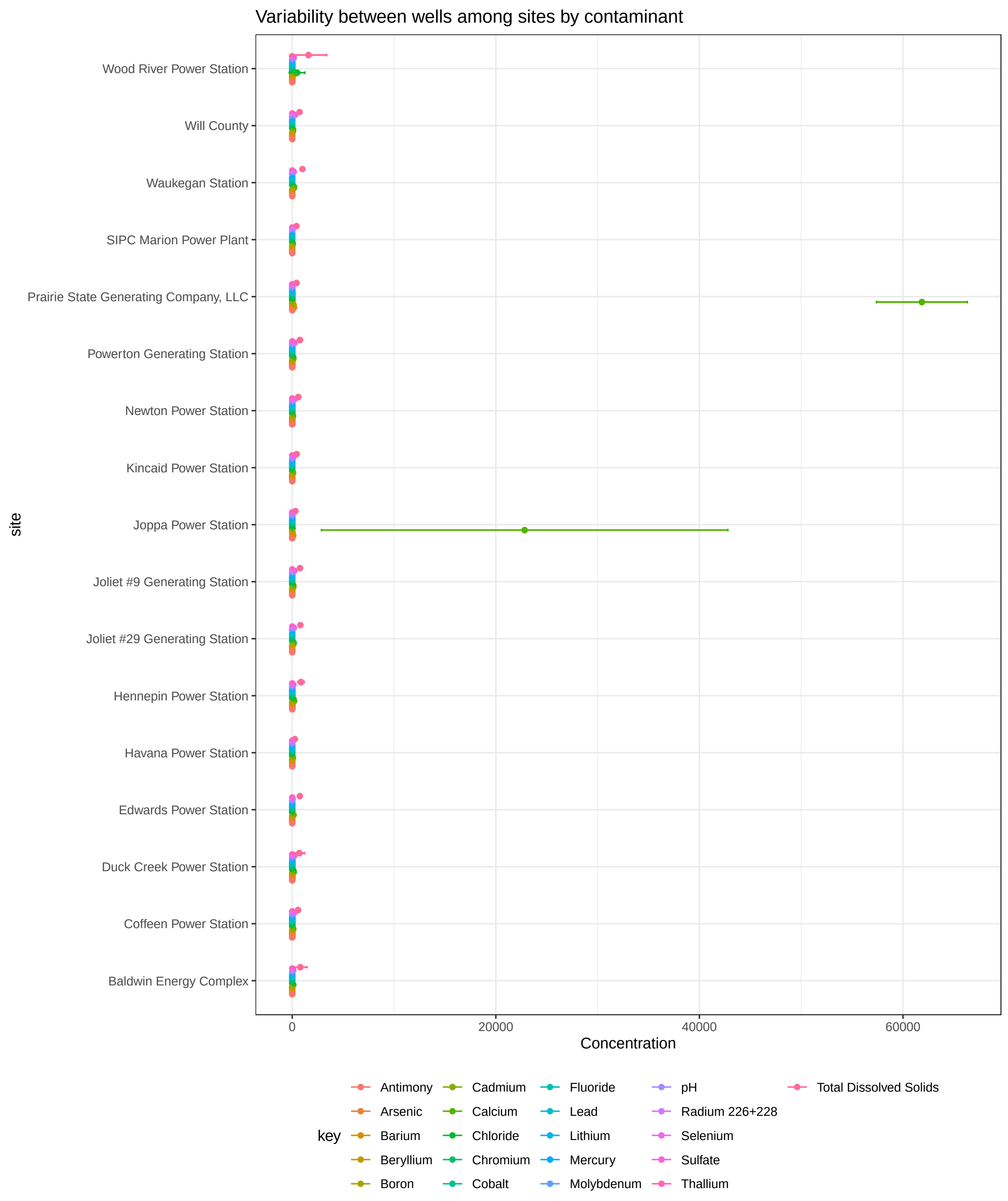
"If you haven't already, you can try some alternative filtering approaches: there might be a lot of contaminants that don't vary that much among the wells, so one thing you could do is find the contaminants with the highest between-well variability. For each chemical you can compute the variance across 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."

```r
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: 1,050 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 G01D    Joppa Po~ Joppa East Ash ~ SI    Upgradi~ Calci~ 3.70e4 22829. 19959.
##  2 G02D    Joppa Po~ Joppa East Ash ~ SI    Upgradi~ Calci~ 3.14e4 22829. 19959.
##  3 G101    Joppa Po~ Joppa Landfill   L     Upgradi~ Calci~ 1.17e1 22829. 19959.
##  4 G03D    Prairie ~ Near Field Land~ L     Upgradi~ Calci~ 5.85e4 61858.  4458.
##  5 MW201   Prairie ~ Near Field Land~ L     Upgradi~ Calci~ 5.78e4 61858.  4458.
##  6 MW203   Prairie ~ Near Field Land~ L     Upgradi~ Calci~ 6.71e4 61858.  4458.
##  7 MW24D   Prairie ~ Near Field Land~ L     Upgradi~ Calci~ 6.40e4 61858.  4458.
##  8 25      Wood Riv~ Wood River West~ SI    Upgradi~ Total~ 1.10e3  1597.  1776.
##  9 31      Wood Riv~ Wood River West~ SI    Upgradi~ Total~ 4.22e3  1597.  1776.
## 10 36      Wood Riv~ Wood River West~ SI    Upgradi~ Total~ 3.72e2  1597.  1776.
## # ... with 1,040 more rows
```

```r
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 84 rows containing missing values (geom_errorbar).
```

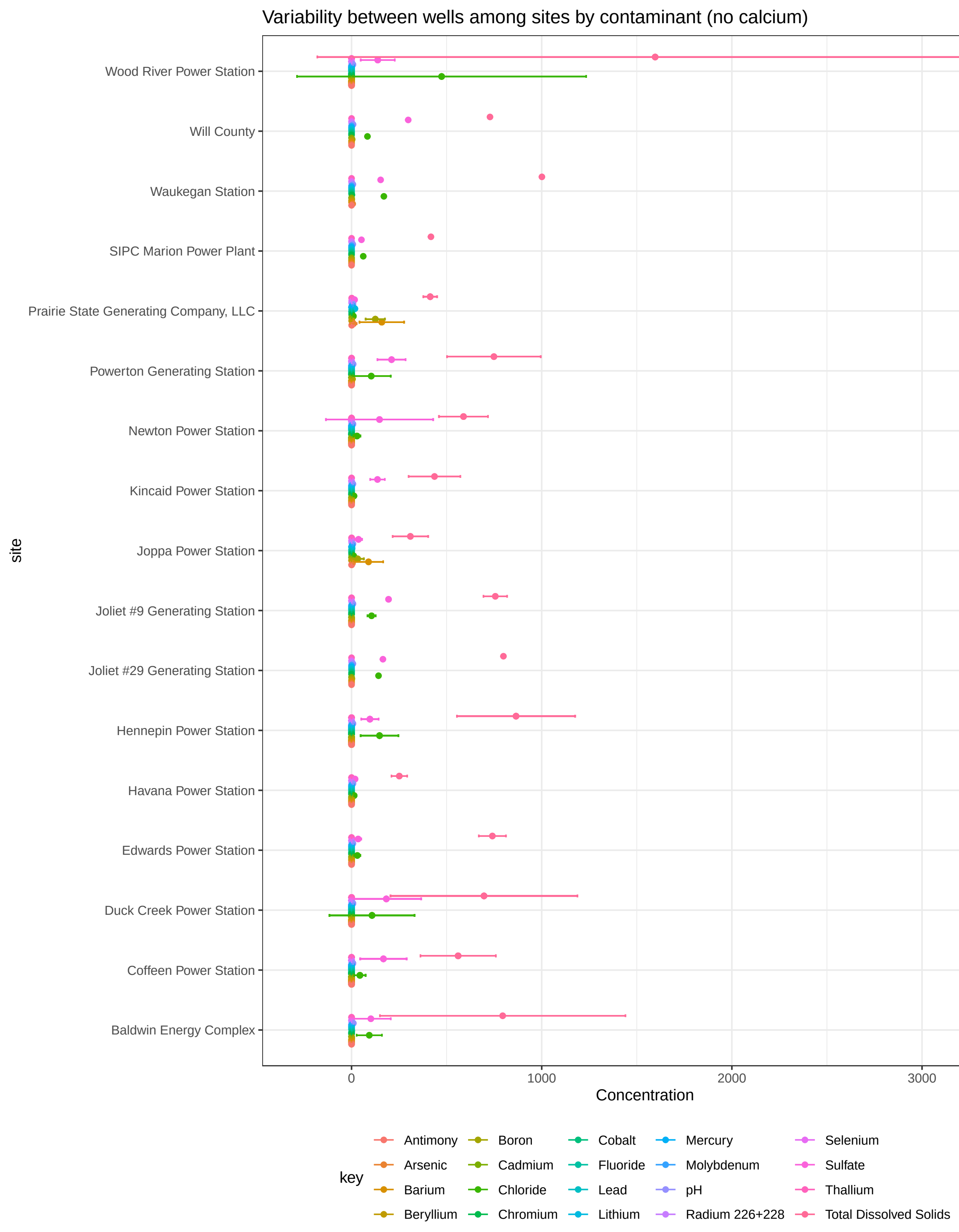


Calcium dominates the entire graph... Remove it...

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

```r
  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 80 rows containing missing values (geom_errorbar).
```



Variability between wells among sites by contaminant (no calcium)
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
site
0
1000
2000
3000
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