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
```

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*.

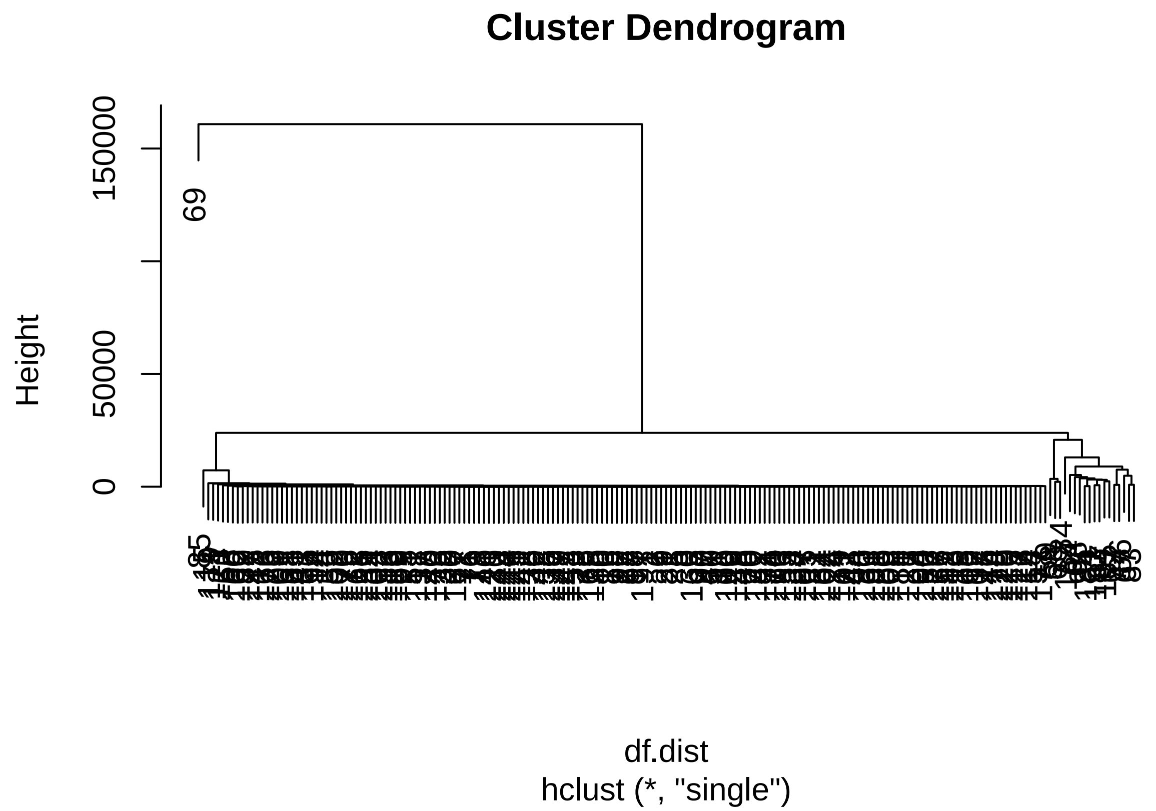
```
hcsingle <- hclust(df.dist, method = "single")
list(hcsingle) # reminds you of properties of the solution, if desired
```

```
## [[1]]
##
## Call:
```

```
## hclust(d = df.dist, method = "single")
##
## Cluster method   : single
## Distance         : euclidean
## Number of objects: 191
```

This creates the solution, and we can look at the dendrogram as:

```
plot(hcsingle)
```



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”.

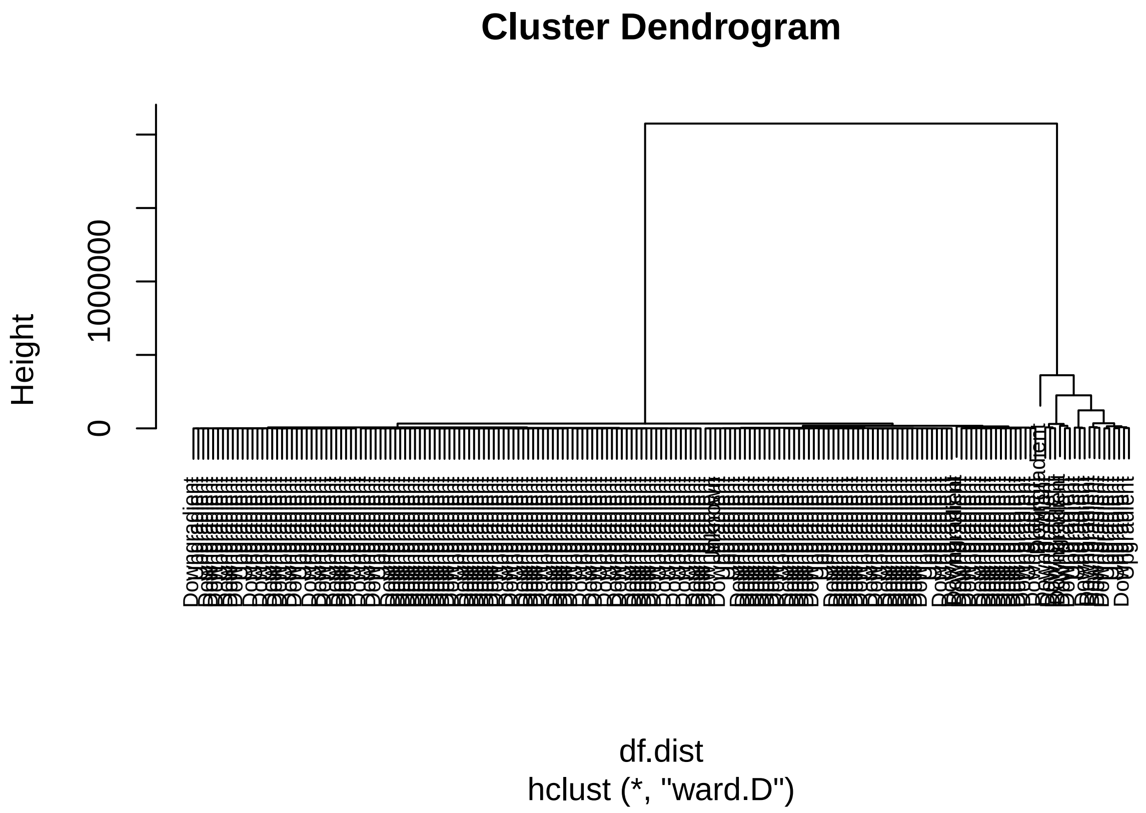
In order to obtain cluster labels, we need to *cut* our dendrograms.

```
singleSol <- (cutree(hcsingle, k = 3)) #cluster labels are numeric, k= # clusters
summary(as.factor(singleSol)) #as factor to get table
```

```
##   1   2   3
## 172  18   1
```

To learn more details about the clusters we found:

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



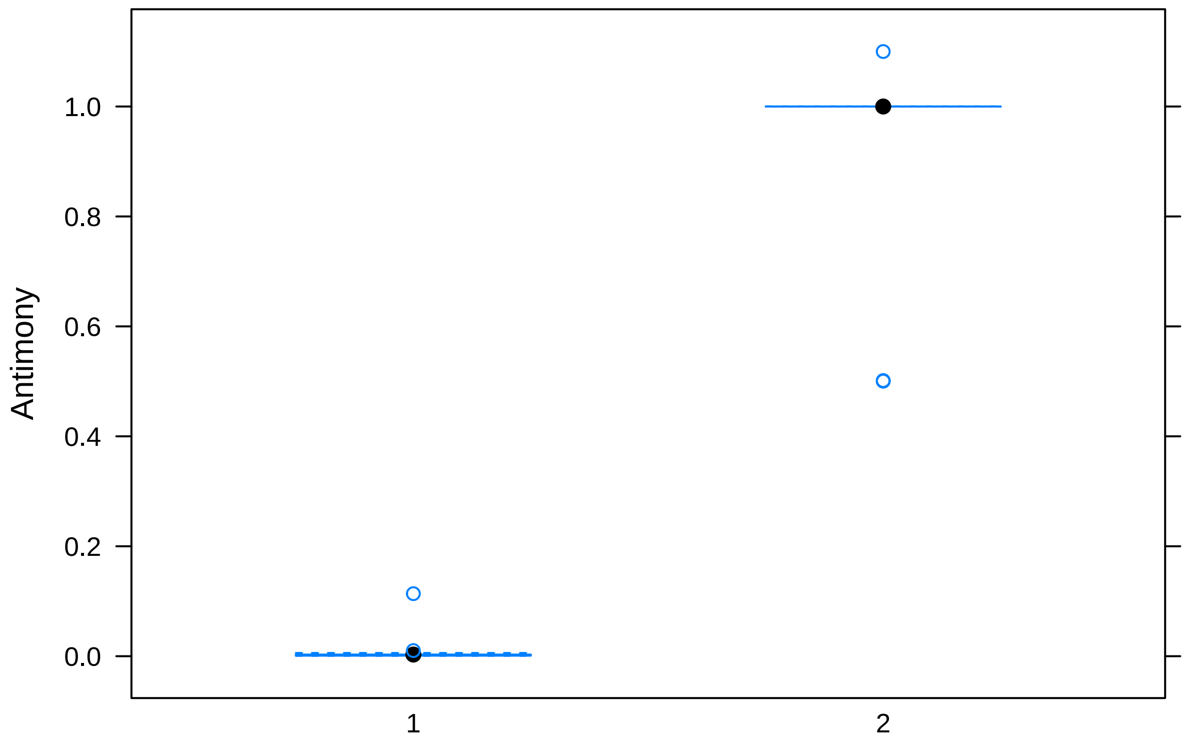
```r
wardSol <- (cutree(hcward, k = 2)) #cluster labels are numeric, k= # clusters
summary(as.factor(wardSol)) #as factor to get table
```

```
##   1   2 
## 172  19
```

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

```
##   wardSol       min    Q1 median    Q3       max        mean         sd   n
## 1       1 0.0007625 0.001  0.003 0.003 0.1137778 0.002903086 0.00858458 172
## 2       2 0.5005000 1.000  1.000 1.000 1.1000000 0.926447368 0.19071691  19
##   missing
## 1       0
## 2       0
```

```r
bwplot(Antimony ~ as.factor(wardSol), data = df1)
```



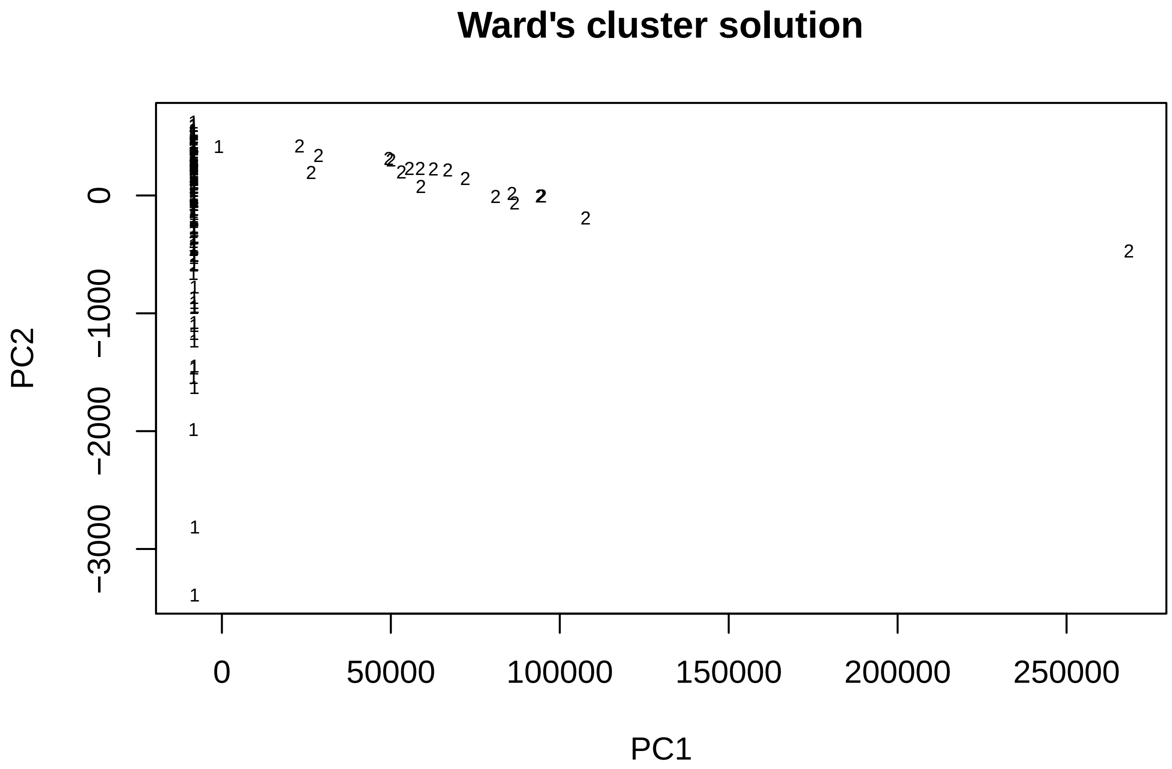
Our cluster sizes are extremely uneven… Our first clsuter has 172 wells and the second has 19.

```
favstats(#chemical here ~ wardSol, data = df1)
bwplot(#chemical here ~ as.factor(wardSol), data = df1)
```

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



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 16, 175
##
## Cluster means:
##     Antimony    Arsenic     Barium  Beryllium      Boron    Cadmium    Calcium
## 1  3.2161104  2.4960121  2.6978935  3.1156481  3.1096817  3.1138065  2.8270214
## 2 -0.2940444 -0.2282068 -0.2466646 -0.2848593 -0.2843138 -0.2846909 -0.2584705
##      Chloride   Chromium     Cobalt    Fluoride       Lead   Lithium    Mercury
## 1 -0.39474887  1.4858331  1.680383 -0.13626181  1.4040106  2.817204  3.2148635
## 2  0.03609133 -0.1358476 -0.153635  0.01245822 -0.1283667 -0.257573 -0.2939304
##   Molybdenum          pH Radium 226+228   Selenium     Sulfate  Thallium
## 1  2.5288750  0.58124207     0.33270030  3.1317800 -0.53202903  3.202237
## 2 -0.2312114 -0.05314213    -0.03041831 -0.2863342  0.04864265 -0.292776
##   Total Dissolved Solids
## 1            -0.46565729
## 2             0.04257438
##
## 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 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 1 2 2 2 2 2
##  [75] 2 2 1 2 2 2 2 2 1 1 2 1 1 2 2 2 2 2 2 1 2 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 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 2 2 2 2 1 1 1
## [186] 2 1 2 2 2 2
##
## Within cluster sum of squares by cluster:
## [1]  792.9099 1210.3005
##  (between_SS / total_SS =  49.8 %)
##
## 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
```

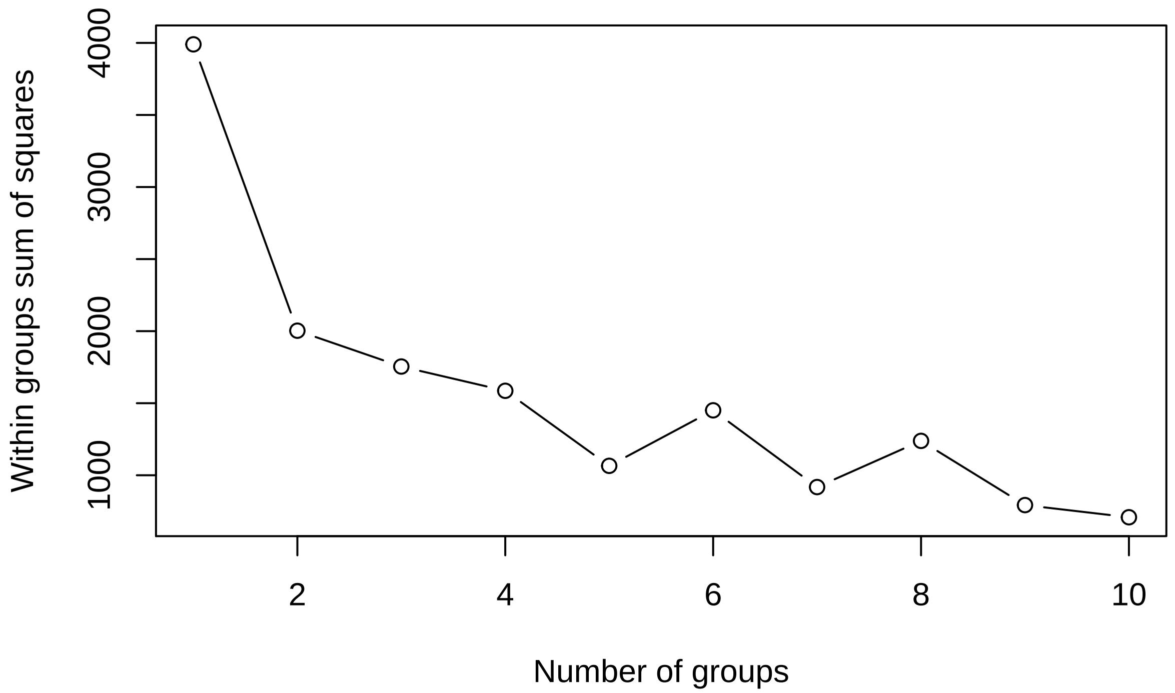
```
##     Antimony    Arsenic     Barium  Beryllium      Boron    Cadmium    Calcium
## 1  3.2161104  2.4960121  2.6978935  3.1156481  3.1096817  3.1138065  2.8270214
## 2 -0.2940444 -0.2282068 -0.2466646 -0.2848593 -0.2843138 -0.2846909 -0.2584705
##      Chloride   Chromium    Cobalt    Fluoride       Lead   Lithium    Mercury
## 1 -0.39474887  1.4858331  1.680383 -0.13626181  1.4040106  2.817204  3.2148635
## 2  0.03609133 -0.1358476 -0.153635  0.01245822 -0.1283667 -0.257573 -0.2939304
##   Molybdenum          pH Radium 226+228   Selenium     Sulfate  Thallium
## 1  2.5288750  0.58124207     0.33270030  3.1317800 -0.53202903  3.202237
## 2 -0.2312114 -0.05314213    -0.03041831 -0.2863342  0.04864265 -0.292776
##   Total Dissolved Solids
## 1            -0.46565729
## 2             0.04257438
```

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  16   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  16
##             2 172   3
```

They seem to match up fairly well!

Alternative Approaches

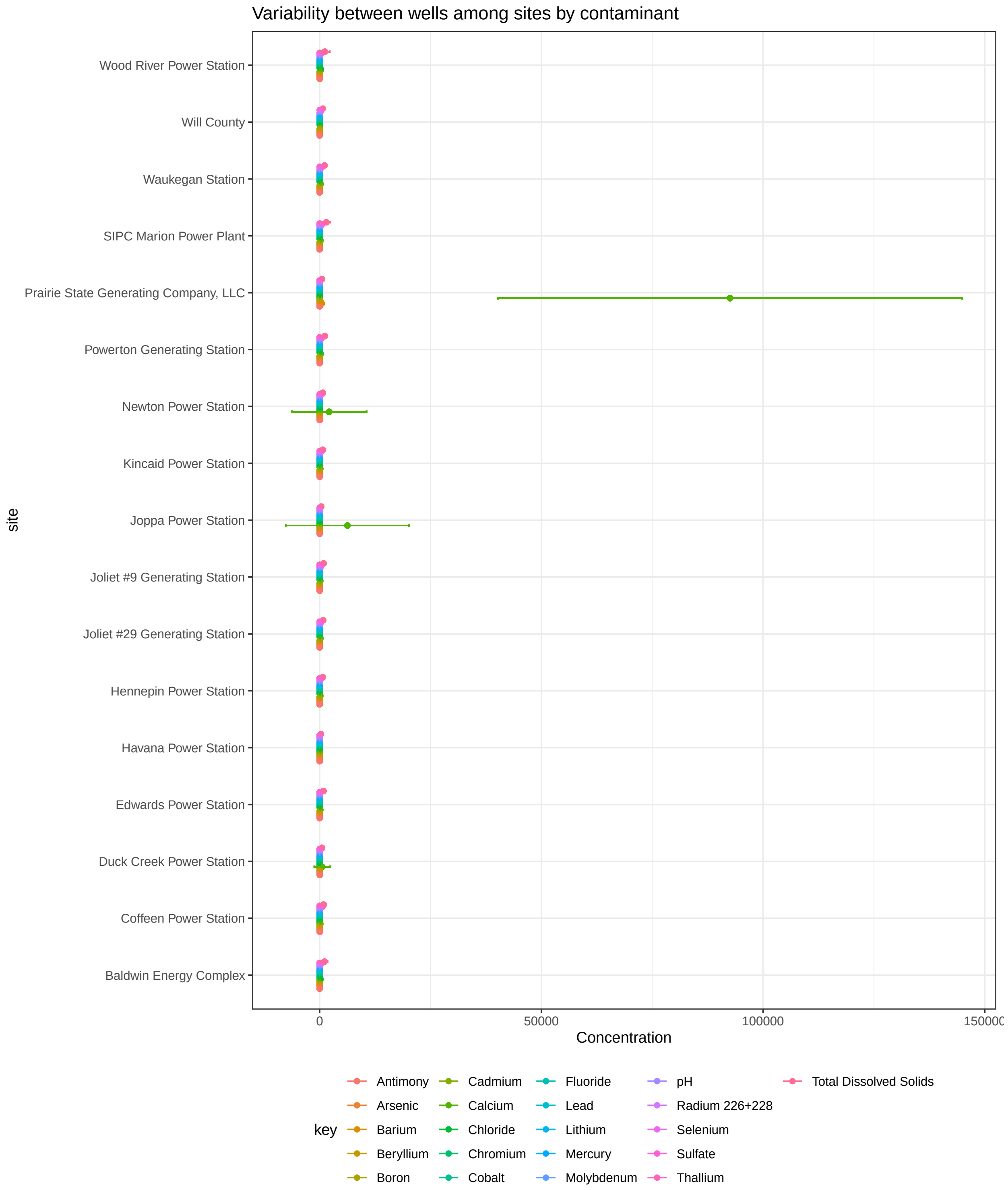
“If you haven’t already, you can try some alternative filtering approaches: there might be a lot of contaminants that don’t vary that much among the wells, so one thing you could do is find the contaminants with the highest between-well variability. For each chemical you can compute the variance across 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: 4,011 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 92542. 52347.
##  2 G04D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 1.03e5 92542. 52347.
##  3 G05D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 7.53e4 92542. 52347.
##  4 G07D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 9.43e4 92542. 52347.
##  5 G08D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 7.10e4 92542. 52347.
##  6 G09D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 2.77e5 92542. 52347.
##  7 G10D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 1.03e5 92542. 52347.
##  8 G13D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 9.51e4 92542. 52347.
##  9 G15D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 1.16e5 92542. 52347.
## 10 G17D    Prairie St~ Near Field Lan~ L     Downgra~ Calc~ 8.94e4 92542. 52347.
## # ... with 4,001 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")
```

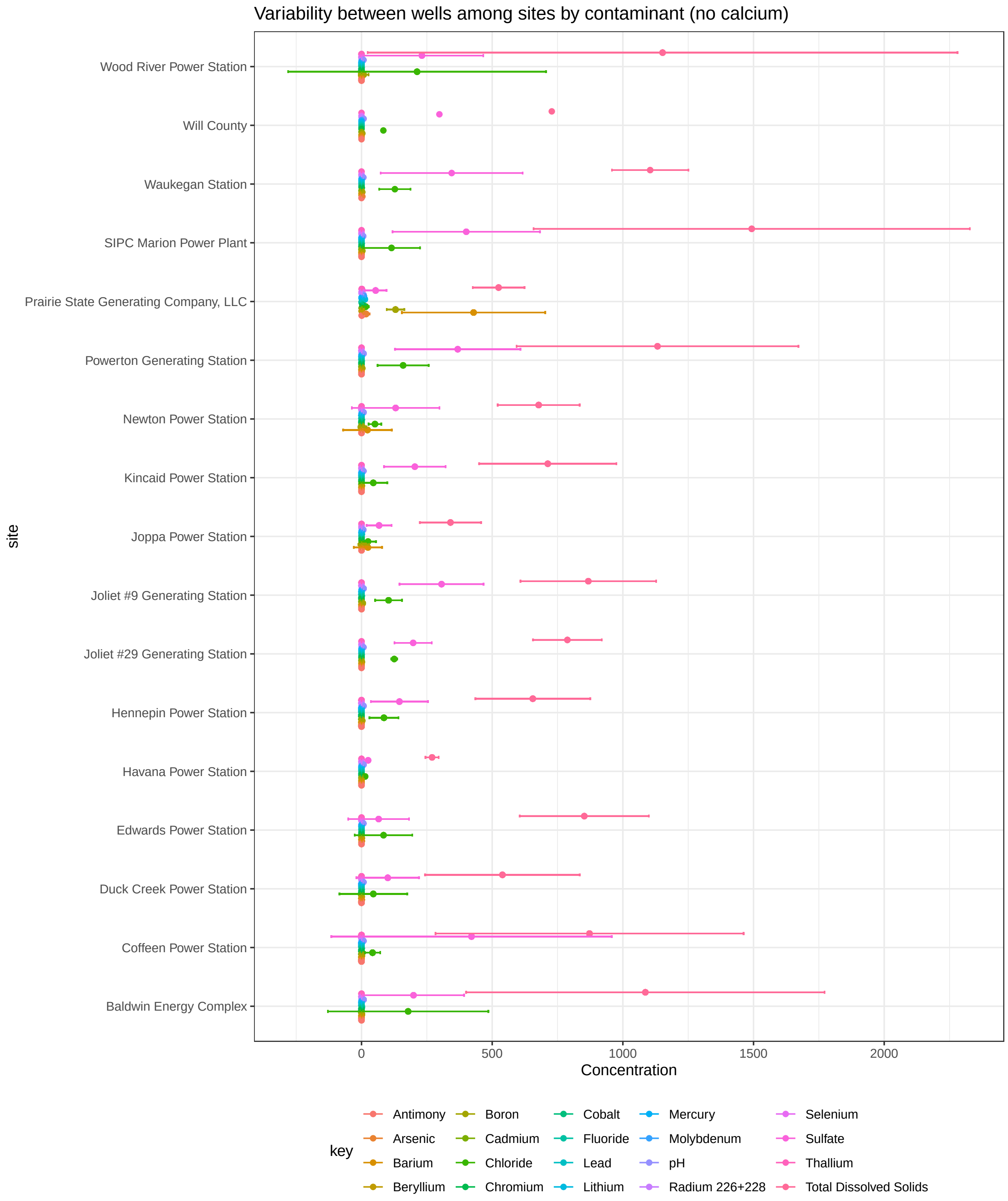
```
## Warning: Removed 21 rows containing missing values (geom_errorbar).
```



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