exploratory

Tony Ni

8/30/2020

```
#Libraries
library(mosaic)
library(tidyverse)
library(usmap)
```

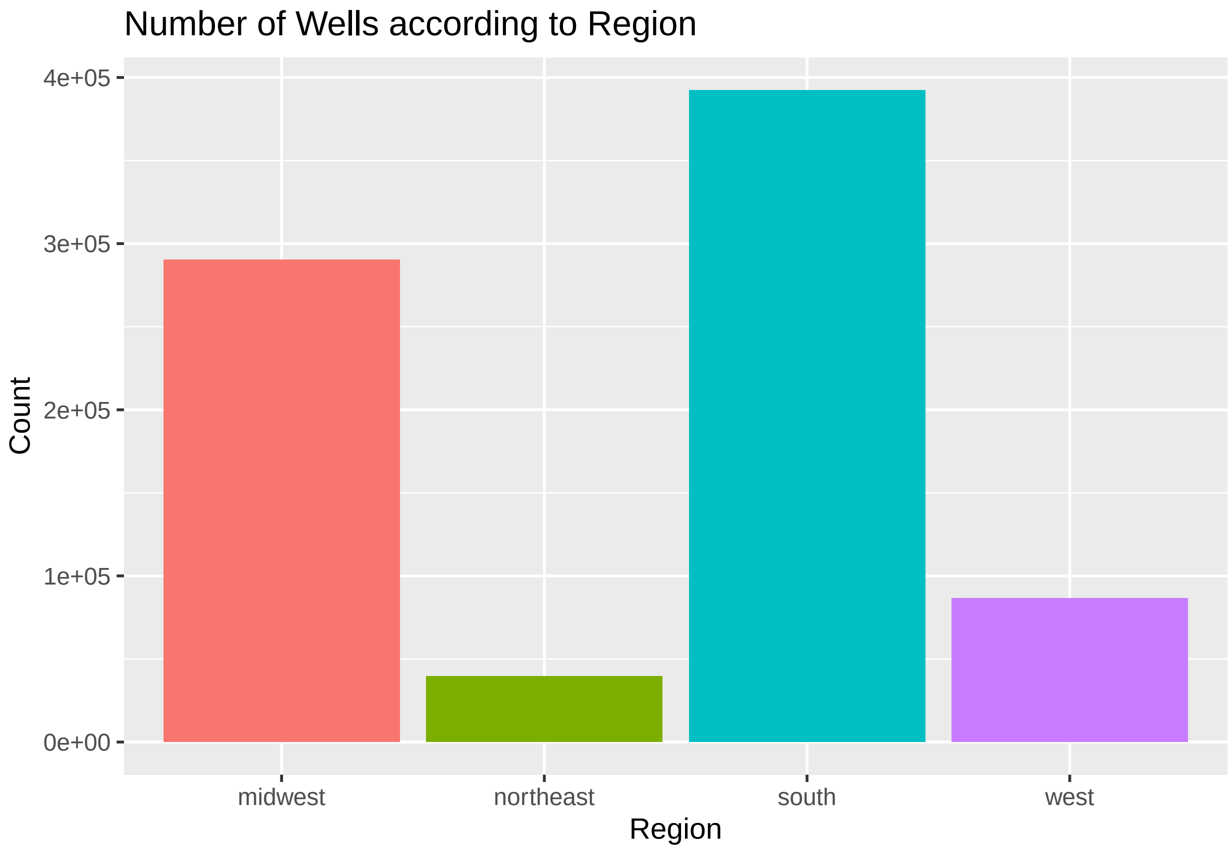
```
#importing in full dataset
import_df <- read_csv("data/chemical_data.csv")
```

```
## Parsed with column specification:
## cols(
##   state = col_character(),
##   site = col_character(),
##   disposal.area = col_character(),
##   type = col_character(),
##   well.id = col_character(),
##   gradient = col_character(),
##   samp.date = col_character(),
##   contaminant = col_character(),
##   measurement.unit = col_character(),
##   below.detection = col_character(),
##   concentration = col_double(),
##   qualifier = col_character(),
##   link = col_character()
## )
```

```
#breaking apart into different datasets for each region
northeast <- import_df %>%
  filter(state %in% c("ME", "NH", "VT", "NY", "PA", "NJ", "MD",
                      "MA", "DE", "RI", "CT")) %>%
  mutate(region = "northeast")

midwest <- import_df %>%
  filter(state %in% c("OH", "IN", "MI", "IL", "WI", "MN", "IA",
                      "MO", "ND", "SD", "NE", "KS"))%>%
  mutate(region = "midwest")

west <- import_df %>%
  filter(state %in% c("WA", "MT", "OR", "ID", "WY", "CA", "NV",
                      "UT", "CO", "AZ", "NM", "AK", "HI")) %>%
  mutate(region = "west")

south <- import_df %>%
  filter(state %in% c("WV", "VA", "KY", "TN", "NC", "SC", "GA",
                      "FL", "MS", "AL", "LA", "AR", "OK", "TX", "PR")) %>%
  mutate(region = "south")

#rejoin them back together for future ref. if needed
full <- list(northeast, midwest, west, south) %>%
  reduce(full_join)
```

```
## Joining, by = c("state", "site", "disposal.area", "type", "well.id", "gradient", "samp.date", "conta
## Joining, by = c("state", "site", "disposal.area", "type", "well.id", "gradient", "samp.date", "conta
## Joining, by = c("state", "site", "disposal.area", "type", "well.id", "gradient", "samp.date", "conta
```

```
ggplot(full, aes(x = region)) +
  geom_bar(aes(fill = region), show.legend = FALSE) +
  ggtitle("Number of Wells according to Region") +
  xlab("Region") +
  ylab("Count")
```



```
midwest_n <- midwest %>%
  group_by(state) %>%
  summarize(n = n()) %>%
  arrange(desc(n))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
northeast_n <- northeast %>%
  group_by(state) %>%
  summarize(n = n()) %>%
  arrange(desc(n))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
south_n <- south %>%
  group_by(state) %>%
  summarize(n = n()) %>%
  arrange(desc(n))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

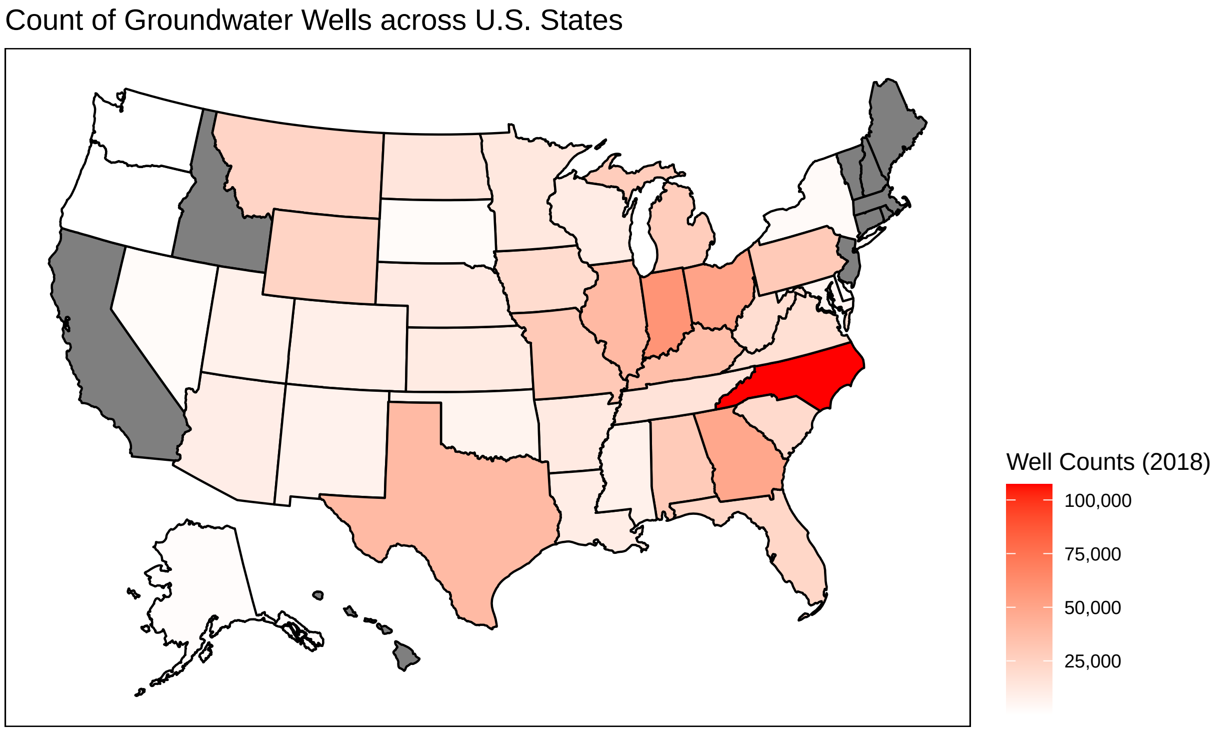
```
west_n <- west %>%
  group_by(state) %>%
  summarize(n = n()) %>%
  arrange(desc(n))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
states_n <- rbind(midwest_n, northeast_n, south_n, west_n)
```

idea: make colored map based on how many wells are in each state

```
state_name <- state.name
state_abb <- state.abb
states_map <- map_data("state")

plot_usmap(data = states_n, values = "n", regions = "states") +
  scale_fill_continuous(low = "white", high = "red",
                        name = "Well Counts (2018)",
                        label = scales::comma) +
  theme(legend.position = "right",
        panel.background = element_rect(color = "black",
                                        fill = "white")) +
  ggtitle("Count of Groundwater Wells across U.S. States")
```



North Carolina has a significant number of wells amongst all states (over 100,000) compared to the next highest which is Indiana with around 58,000.

Let’s focus in on North Carolina only for now!

```
NC <- south %>%
  filter(state %in% "NC")

#count of sites
NC %>%
  group_by(site) %>%
  summarize(n = n()) %>%
  arrange(desc(n))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 13 x 2
##    site                                   n
##    <chr>                              <int>
##  1 "L.V. Sutton Energy Complex"       15683
##  2 "Belews Creek Steam Station"       13414
##  3 "Cliffside Steam Station"          13362
##  4 "Roxboro Steam Electric Plant"     10320
##  5 "Allen Steam Station"               9938
##  6 "Buck Steam Station"                9813
##  7 "Dan River Steam Station"           7885
##  8 "Mayo Steam Electric Plant"         7560
##  9 "H.F. Lee Energy Complex"           6120
## 10 "Marshall Steam Station"             6047
## 11 "Asheville Steam Electric Plant"     4115
## 12 "W.H. Weatherspoon Power Plant"      2520
## 13 "Brickhaven No. 2 Mine Tract \"A\""   357
```

There are 13 different “sites” in which the wells can belong to.

```
#count of disposal.area
NC %>%
  group_by(disposal.area) %>%
  summarize(n = n()) %>%
  arrange(desc(n))
```

```
## ‘summarise()‘ ungrouping output (override with ‘.groups‘ argument)
```

```
## # A tibble: 22 x 2
##    disposal.area                                                               n
##    <chr>                                                                   <int>
##  1 Active Ash Basin                                                        15434
##  2 CCP Landfill                                                            11572
##  3 1971 and 1984 Ash Basins                                                10620
##  4 Active Ash Basin, Retired Ash Basin, Retired Ash Basin Landfill          9938
##  5 CCR Multiunit 2 (West Ash Basin, East and West FGD Settling Ponds, FGD~  6120
##  6 Active Ash Basin and Industrial Landfill No. 1                           6047
##  7 Craig Road Landfill                                                      5156
##  8 Primary Pond (Ash Basin 2), Secondary Pond (Ash Basin 3)                 5003
##  9 Additional Primary Pond (Ash Basin 1)                                    4810
## 10 CCR Multiunit 1 (East Ash Pond, Industrial Landfill)                     4200
## # ... with 12 more rows
```

Within each well, there are multiple disposal areas also (total count of 22).

```
#count of gradient
NC %>%
  group_by(gradient) %>%
  summarize(n = n()) %>%
  arrange(desc(n))
```

```
## ‘summarise()‘ ungrouping output (override with ‘.groups‘ argument)
```

```
## # A tibble: 4 x 2
##   gradient                       n
##   <chr>                      <int>
## 1 Downgradient               92169
## 2 Upgradient                 14063
## 3 Unknown                      629
## 4 Downgradient/Crossgradient   273
```

In the case for the NC wells, there are 92.169 downgradient wells and 14,063 upgradient wells.

Dangerous Toxins in Coal Ash

Some of the most dangerous contaminants often found in coal ash include: arsenic, lead, mercury, cadmium, chromium, and selenium (https://www.psr.org/wp-content/uploads/2018/05/coal-ash-toxics.pdf)

Are there different rates of censoring for different contaminants?

```
NC_subset <- NC %>%
  filter(contaminant %in% c("Arsenic, dissolved", "Arsenic, total",
                            "Lead, total", "Mercury, total",
                            "Cadmium, dissolved", "Cadmium, total",
                            "Chromium, total", "Selenium, Dissolved",
                            "Selenium, Total")) %>%
  #exclude crossgradient and unknown
  filter(gradient %in% c("Upgradient", "Downgradient"))

NC_subset2 <- NC_subset %>%
  group_by(contaminant, gradient, measurement.unit, below.detection) %>%
  summarize(n = n())
```

```
## ‘summarise()‘ regrouping output by 'contaminant', 'gradient', 'measurement.unit' (override with ‘.gr
```

```
NC_subset3 <- NC_subset2[-c(27), ] #removing strange sole observation
```

```
## Warning: The ‘i‘ argument of ‘‘[.tbl_df‘()‘ must lie in [-rows, 0] if negative, as of tibble 3.0.0.
## Use ‘NA_integer_‘ as row index to obtain a row full of ‘NA‘ values.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_warnings()‘ to see where this warning was generated.
```

```
NC_subset4 <- NC_subset3 %>%
  group_by(contaminant, gradient, measurement.unit) %>%
  summarize(prop = n/(sum(n))) %>%
  mutate(below.detection = case_when(
    row_number() %% 2 == 1 ~ "<", #odd
    row_number() %% 2 == 0 ~ "NA")) %>% #even
  filter(below.detection %in% "<") %>%
  arrange(order(contaminant))
```

```
## ‘summarise()‘ regrouping output by 'contaminant', 'gradient', 'measurement.unit' (override with ‘.gr
```

```
knitr::kable(NC_subset4)
```

contaminant	gradient	measurement.unit	prop	below.detection
Arsenic, total	Downgradient	ug/l	0.1235191	<
Arsenic, total	Upgradient	ug/l	0.2423803	<
Cadmium, total	Downgradient	ug/l	0.3968846	<
Cadmium, total	Upgradient	ug/l	0.4702467	<
Chromium, total	Downgradient	ug/l	0.0976305	<
Chromium, total	Upgradient	ug/l	0.1567489	<
Lead, total	Downgradient	ug/l	0.3394032	<
Lead, total	Upgradient	ug/l	0.3889695	<
Mercury, total	Downgradient	ug/l	0.4429574	<
Mercury, total	Upgradient	mg/l	1.0000000	<
Mercury, total	Upgradient	ug/l	0.4912791	<

Generally, when considering only the wells that have measurements below detection – it seems like a higher proportion of the upgradient wells have measurements below detection when compared to their downgradient counterparts.

Ideally, we would like to know the average level of contamination (for a contaminant) with regards to upgradient and downgradient wells. However, due to the the proportions of measurements which are below the limit of detection being so high – it may pose challenges in our endeavor.

We could try to calculate averages without accounting for censoring to see what happens and then applying our methods and see if there are any differences. We have no way of knowing what the true averages will be due to so many being below the limit of detection – however, we can definitely look to see if there are differences.