IDW model for short-term and long-term sensors

Jordan Wingenroth

May 31, 2018

```
library(tidyverse)
library(readxl)
library(lubridate)
library(maps)
library(gstat)
library(animation)
library(gganimate)
library(sp)
library(knitr)
library(raster)
```

geographic data setup

It seemed to make the most sense to load the pm 2.5 data from longterm sensors at the same time since the lat/long was included in the same table. The pm 2.5 dataset was a large file (10 MB) since it included all October data for the entire state, so I filtered it to only include sites within ~1 deg of the centerpoint of sampling sites in Excel prior to adding to the project.

```
st_sensors <- read_csv(file = "../data/aq_sensors.csv")
```

```
## Parsed with column specification:
## cols(
##   name = col_character(),
##   `sensor class` = col_character(),
##   latitude = col_double(),
##   longitude = col_double()
## )
```

```
st_sensors <- filter(st_sensors, `sensor class` == "short_term")

long_term_pm25 <- read_csv("../data/long_term_pm25.csv")
```

```
## Parsed with column specification:
## cols(
##   site = col_integer(),
##   monitor = col_integer(),
##   date = col_character(),
##   start_hour = col_integer(),
##   value = col_integer(),
##   variable = col_character(),
##   units = col_character(),
##   quality = col_integer(),
##   prelim = col_character(),
##   name = col_character(),
##   latitude = col_double(),
##   longitude = col_double(),
##   obs_type = col_character(),
##   monitoring_id = col_character(),
```

```
##   flag = col_character(),
##   time = col_character()
## )
```

```
lt_sensors <-long_term_pm25 %>%
  distinct(name, .keep_all = TRUE) %>%
  transmute(name, "sensor class" = "long_term", latitude, longitude)

farms <- read_csv(file = "../data/farm_data.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   X1 = col_integer(),
##   number = col_integer(),
##   address = col_character(),
##   Tiger = col_character(),
##   X4 = col_character(),
##   X5 = col_character(),
##   long = col_double(),
##   lat = col_double(),
##   X7 = col_double(),
##   X8 = col_character(),
##   optional = col_logical()
## )
```

```
hysplit <- readxl::read_xlsx(path = "../data/HYSPLIT_data.xlsx")

all(sort(hysplit$Address)==sort(farms$address))
```

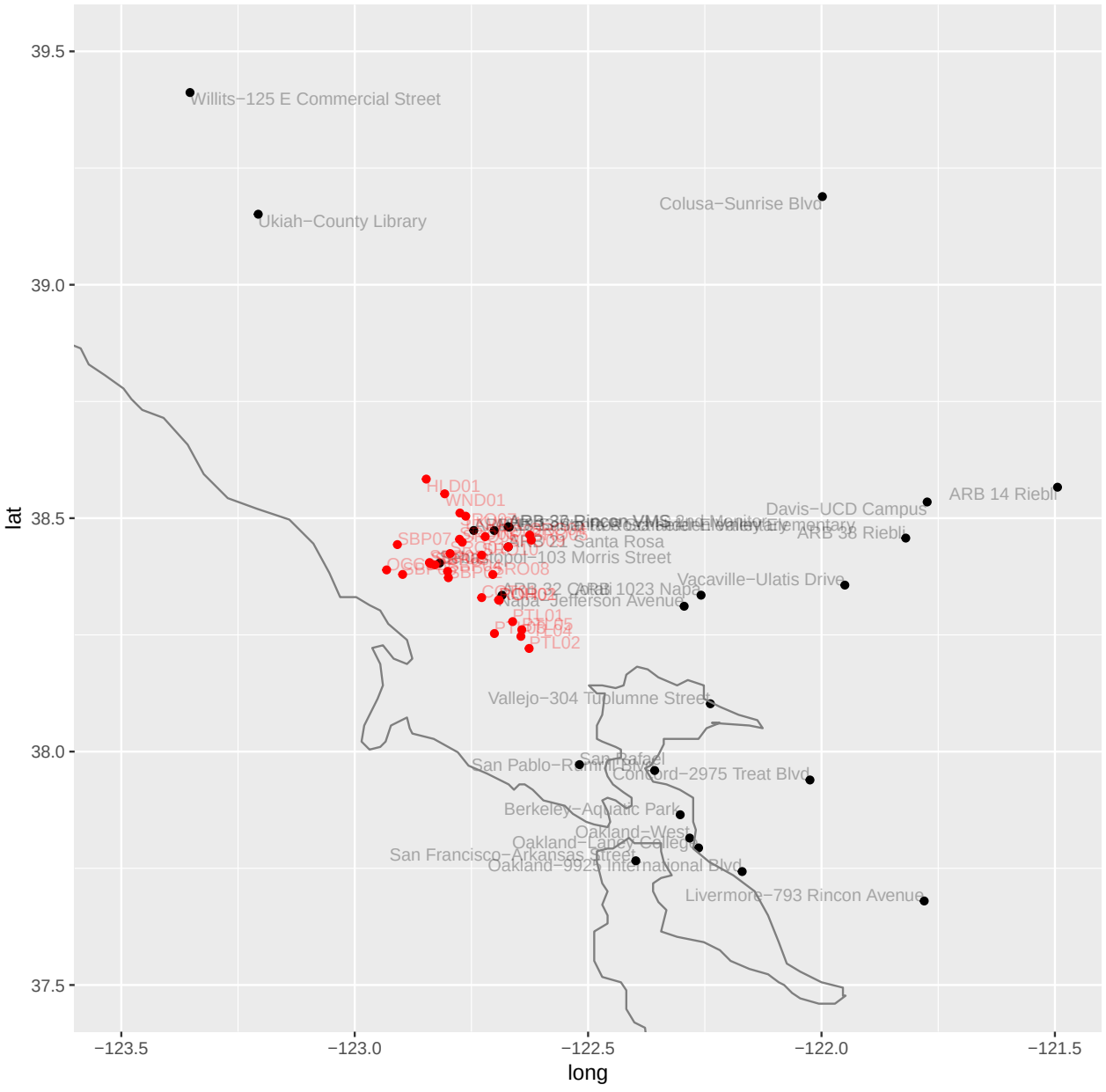
```
## [1] TRUE
```

```
#farm

farms <- left_join(farms, hysplit, by = c("address" = "Address"))

farms <- farms %>%
  dplyr::select(address, Key, long, lat,
                `Exposure (raw, final model, normalized)`,
                `Exposure (smoothed, final model, normalized)`,
                `Rank (raw)`, `Rank (smoothed)`)

rm(list = "hysplit")

#aq_sensors

aq_sensors <- rbind(lt_sensors, st_sensors)
rm(list = c("lt_sensors", "st_sensors"))
aq_sensors <- rename(aq_sensors, lat = latitude, long = longitude)
```

map with farm sites and aq sensors

```
ggplot(NULL, aes(long, lat)) +
  geom_point(data = aq_sensors) +
```

```
  geom_text(data = aq_sensors, color = "black", alpha = .3, aes(label = name),
            hjust = "inward", vjust = "inward", size = 3) +
  geom_point(data = farms, color = "red") +
  geom_text(data = farms, color = "red", alpha = .3, aes(label = Key),
            hjust = "inward", vjust = "inward", size = 3) +
  borders("state", "California") +
  coord_fixed(xlim = c(-123.5,-121.5), ylim = c(37.5, 39.5))
```



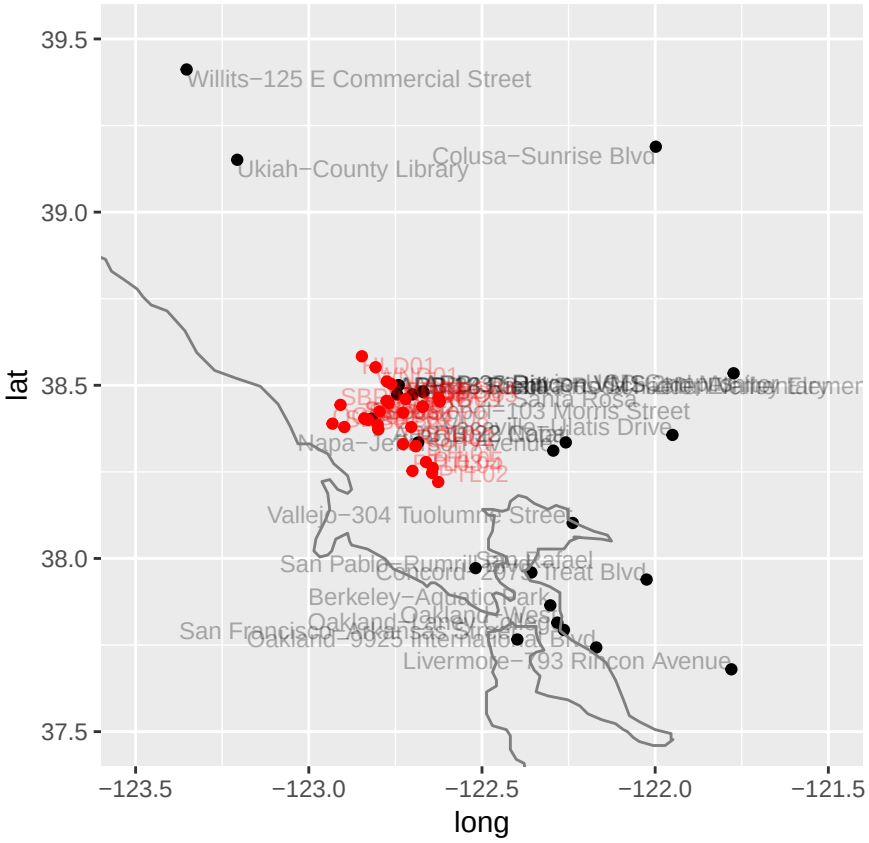
We're lacking sensors to the northwest of the sites, but we have a good number to the southeast (Vacaville, Vallejo, Berkeley, SF, Oakland).

ARB 14 & 38 Riebli sites have incorrect GPS data???

#fix Riebli data

```
#values from Google Maps
aq_sensors[grep("Riebli", aq_sensors$name),]$lat <- 38.500
aq_sensors[grep("Riebli", aq_sensors$name),]$long <- -122.741
```

```
ggplot(NULL, aes(long, lat)) +
  geom_point(data = aq_sensors) +
  geom_text(data = aq_sensors, color = "black", alpha = .3, aes(label = name),
            hjust = "inward", vjust = "inward", size = 3) +
  geom_point(data = farms, color = "red") +
  geom_text(data = farms, color = "red", alpha = .3, aes(label = Key),
            hjust = "inward", vjust = "inward", size = 3) +
  borders("state", "California") +
  coord_fixed(xlim = c(-123.5,-121.5), ylim = c(37.5, 39.5))
```



PM 2.5 data setup

```
st_files <- list.files("../data/short_term_pm25", full.names = TRUE)
short_term_pm25 <- lapply(st_files, function(x) read_xlsx(x))
names(short_term_pm25) <- str_sub(st_files, 42, -20)
for (i in 1:length(short_term_pm25)) {

  short_term_pm25[[i]] <- mutate(short_term_pm25[[i]],
                                 name = names(short_term_pm25[i]))

}
```

```
short_term_pm25 <- do.call(rbind, short_term_pm25)
short_term_pm25 <- dplyr::select(short_term_pm25, name, everything())

sort(unique(short_term_pm25$name))==sort(aq_sensors[aq_sensors$`sensor class`=="short_term",]$name)
```

```
## [1]  TRUE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE FALSE
```

Some of the short term sensors have different, but easily matchable names.

```
#short-term times

short_term_pm25 <- short_term_pm25 %>%
  mutate(datetime = `Date/Time/` + 3600*hour(PST))


#long-term times

long_term_pm25 <- long_term_pm25 %>%
  mutate(date = as.POSIXct(long_term_pm25$date, format = "%m/%d/%Y"), datetime = date + 3600*start_hour

#match sites to aq sensors by key

aq_sensors$key <- str_sub(aq_sensors$name, end = 10)
long_term_pm25$key <- str_sub(long_term_pm25$name, end = 10)
short_term_pm25$key <- str_sub(short_term_pm25$name, end = 10)

sum(sort(unique(aq_sensors$key))==sort(c(unique(long_term_pm25$key), unique(short_term_pm25$key))))
```

```
## [1] 26
```

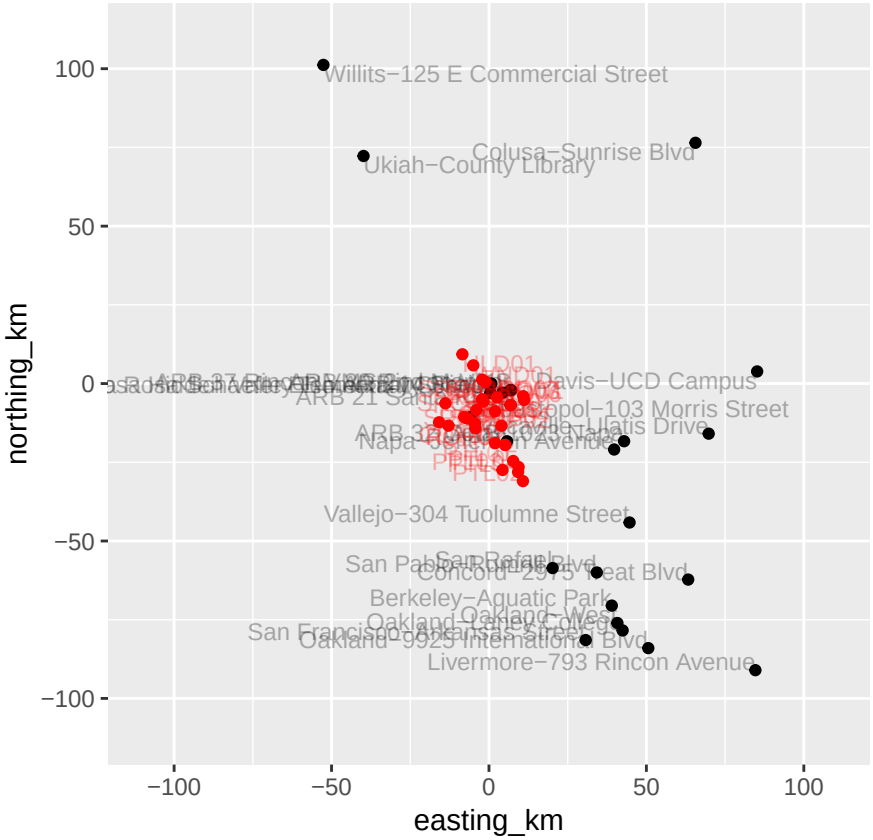
```
#pull out and match up essential columns for purpose of rowbinding and joining

t1<-short_term_pm25 %>%
  dplyr::select(key, name, datetime, value = ConcHr)

t2<-long_term_pm25 %>%
  dplyr::select(key, name, datetime, value)

#rowbind and join to gps data

pm25 <- rbind(t1, t2)

pm25 <- pm25 %>%
  left_join(aq_sensors, "key") %>%
  dplyr::select(-name.y) %>%
  rename(name = name.x)

rm(list = c("aq_sensors","long_term_pm25","short_term_pm25","t1","t2"))

#Filter out negative values and zero values as they could be errors and should be bounded by small numm
#Also filter out some Vacaville values that appear to be errors (>900 ug/m3)

pm25 <- pm25 %>%
  filter(value>=0, value<900)
```

```
## censor overfit data

pm25 <- pm25 %>%
  filter(key!="ARB 33 San")
```

convert lat/long to kilometers for all data

Data from http://www.csgnetwork.com/degreelenllavcalc.html

```
farms<- farms %>%
  mutate(northing_km = 111.005*(lat-38.5), easting_km = 87.233*(long+122.75))

pm25 <- pm25 %>%
  mutate(northing_km = 111.005*(lat-38.5), easting_km = 87.233*(long+122.75))

ggplot(NULL, aes(easting_km, northing_km)) +
  geom_point(data = distinct(pm25,long,lat,.keep_all=TRUE)) +
  geom_text(data = distinct(pm25,long,lat,.keep_all=TRUE), color = "black", alpha = .3, aes(label = nam
  geom_point(data = farms, color = "red") +
  geom_text(data = farms, color = "red", alpha = .3, aes(label = Key), hjust = "inward", vjust = "inwar
  coord_fixed(xlim=c(-110,110),ylim=c(-110,110))
```

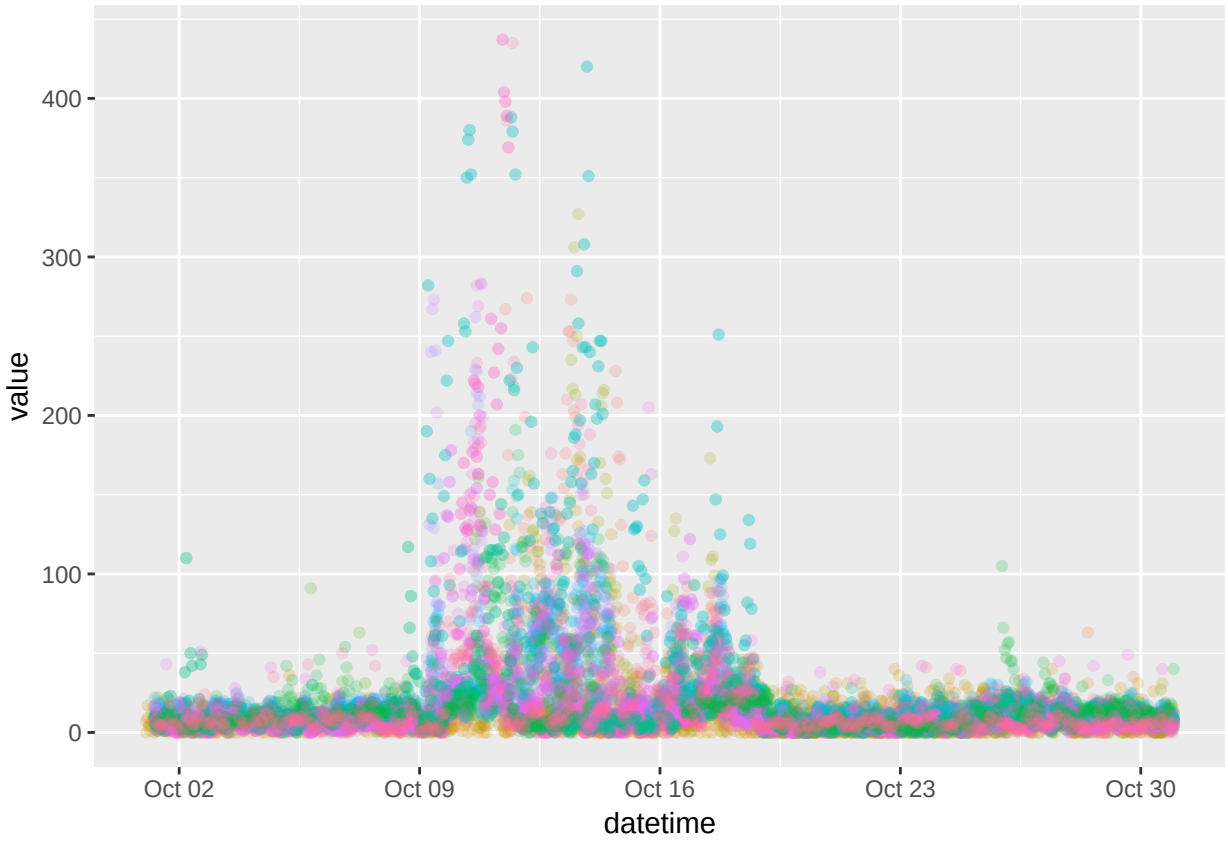


Q for Vanessa: any idea as to where the epicenter of burned industrial facilities was? That would have some significance for our choice of GPS location.

Currently using Larkfield-Wikiup for convenience (quarter-degree lat/long)

graphs of sensor data

```
pm25 %>%
  filter(datetime>as.Date("2017-10-1"), datetime<as.Date("2017-10-31")) %>%
  group_by(name) %>%
  ggplot(aes(datetime, value, color = name)) +
  geom_point(alpha = .2) +
  theme(legend.position = "none")
```

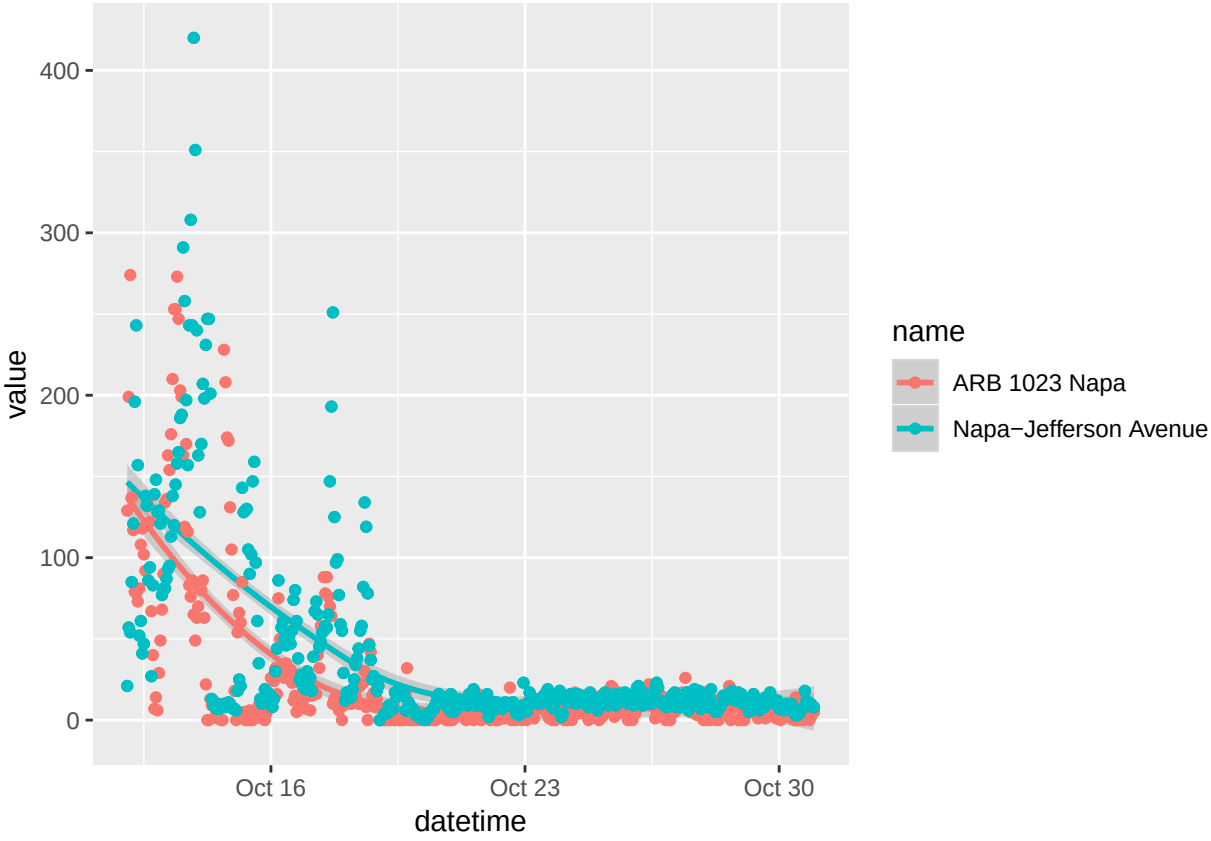


From this, I've arrived at the conclusion that Oct 8 - Oct 20 is a good span for the model run in order to capture the most serious part of the fire.

The only long term and short term sensors in close proximity were the Napa ones. Let's compare:

```
#Set to start on Oct 12 because that's when the short term Napa sensor was placed
pm25 %>%
  filter(datetime>as.Date("2017-10-12"), datetime<as.Date("2017-10-31"), str_detect(name, "Nap")) %>%
  group_by(name) %>%
  ggplot(aes(datetime, value, color = name)) +
  geom_smooth() +
  geom_point()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Reasonably similar, especially given their slight geographic distance.

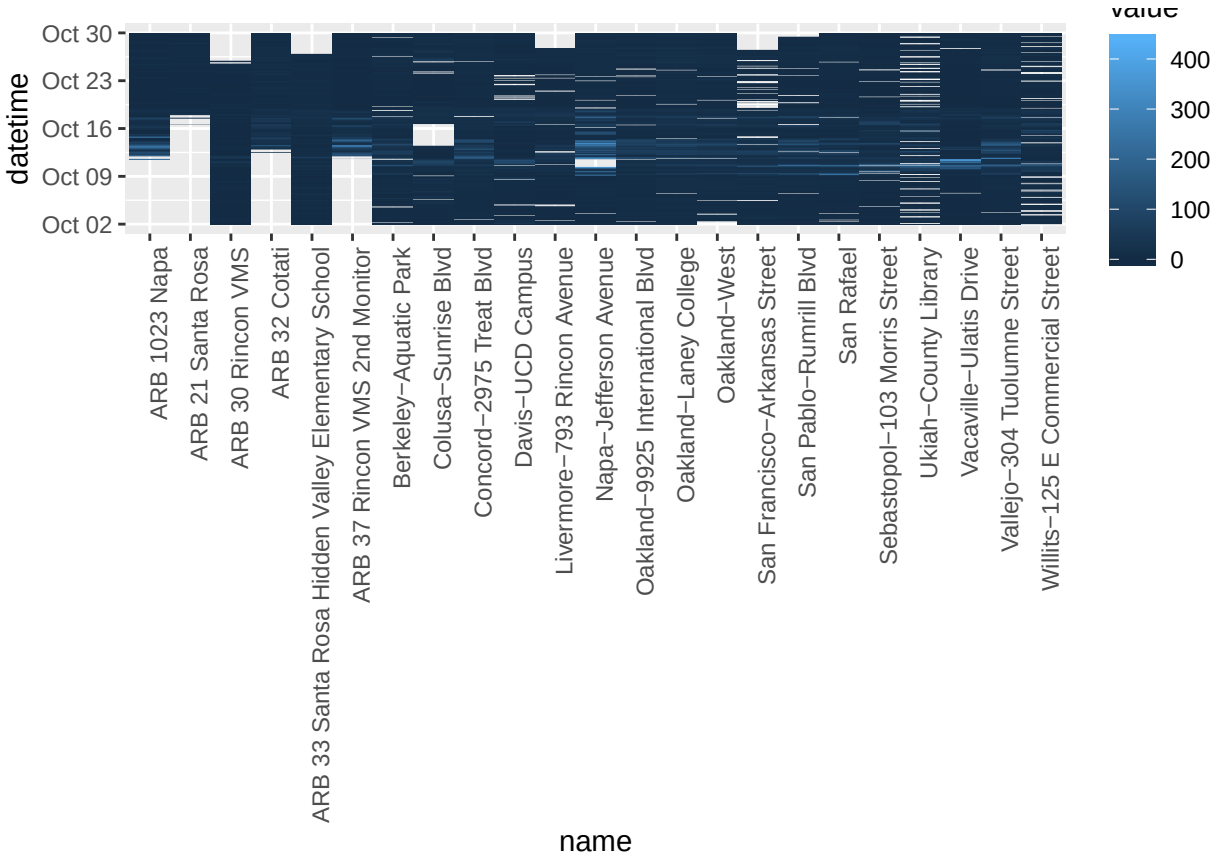
stats on October sensor data

```
pm25 %>%
  filter(datetime>as.Date("2017-10-1"), datetime<as.Date("2017-10-31")) %>%
  group_by(name) %>%
  summarize("n of measurements" = n(), mean = mean(value), min = min(value), Q1 = quantile(value, .25),
  arrange(desc(mean))
```

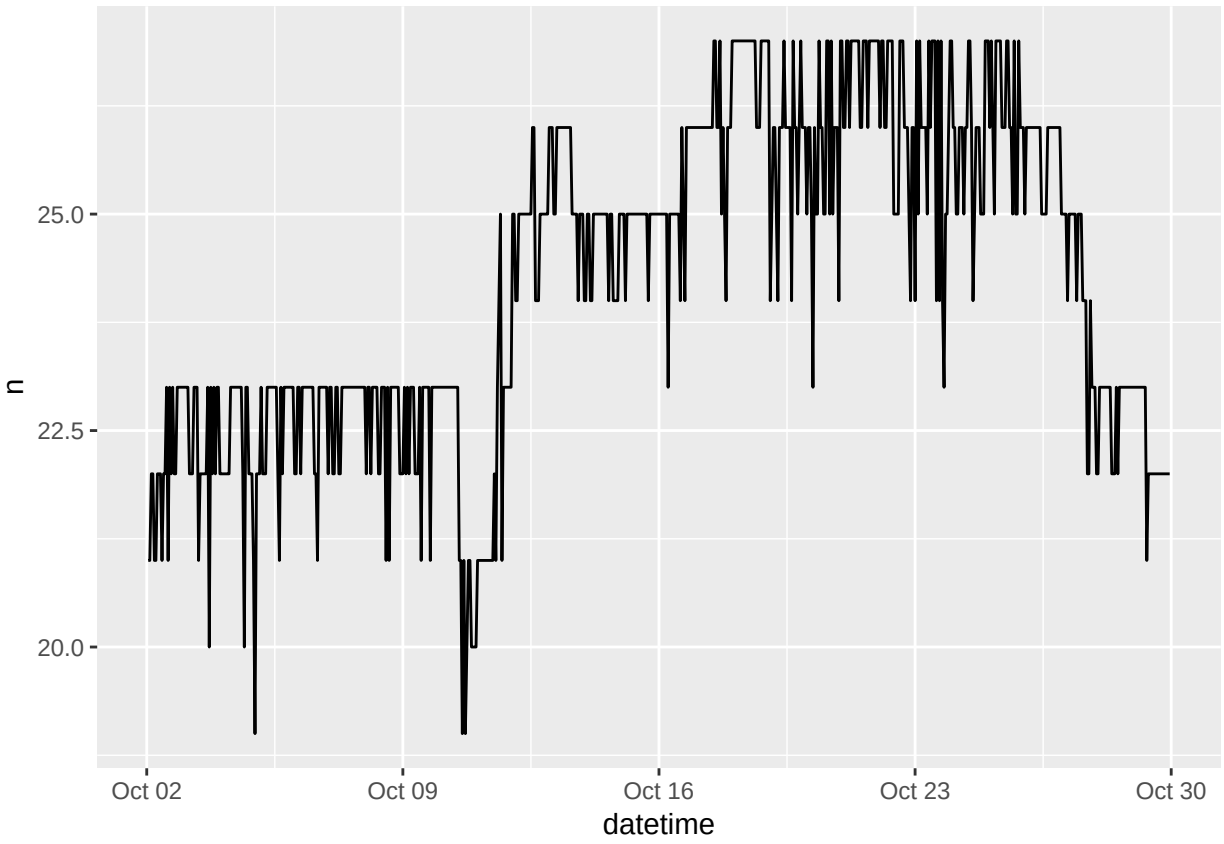
```
## # A tibble: 24 x 8
##    name              `n of measuremen~  mean   min    Q1 median    Q3   max
##    <chr>                         <int> <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>
##  1 Napa-Jefferson ~               1360  37.2     0     8   11      21   420
##  2 ARB 1023 Napa                   461  26.6     0     3    7      17   386
##  3 ARB 37 Rincon V~                458  24.4     0     2    7      19   327
##  4 Vallejo-304 Tuo~                709  21.8     1     7   11      18   435
##  5 Oakland-Laney C~                708  20.6     0    10   15      21   135
##  6 San Pablo-Rumri~                667  19.8     1     7   11      17   241
##  7 Concord-2975 Tr~                709  19.6     3     9   12      18   218
##  8 Oakland-West                    675  19.2     0     9   14      21   123
##  9 Ukiah-County Li~                640  18.5     0     4    7      20   282
## 10 ARB 32 Cotati                   436  17.9     0     6   13.5    23   172
## # ... with 14 more rows
```

sensor data coverage in October

```
#going to go with a range of 10/2-10/30 because sensors are running more or less dependably on that spa

pm25 %>%
  filter(datetime>as.Date("2017-10-2"), datetime<as.Date("2017-10-30")) %>%
  group_by(name) %>%
  ggplot(aes(x = name, y = datetime)) +
  geom_raster(aes(fill = value)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```




```
pm25 %>%
  filter(datetime>as.Date("2017-10-2"), datetime<as.Date("2017-10-30")) %>%
  group_by(datetime) %>%
  summarise(n = n()) %>%
  ggplot(aes(datetime, n)) +
  geom_line()
```



model function

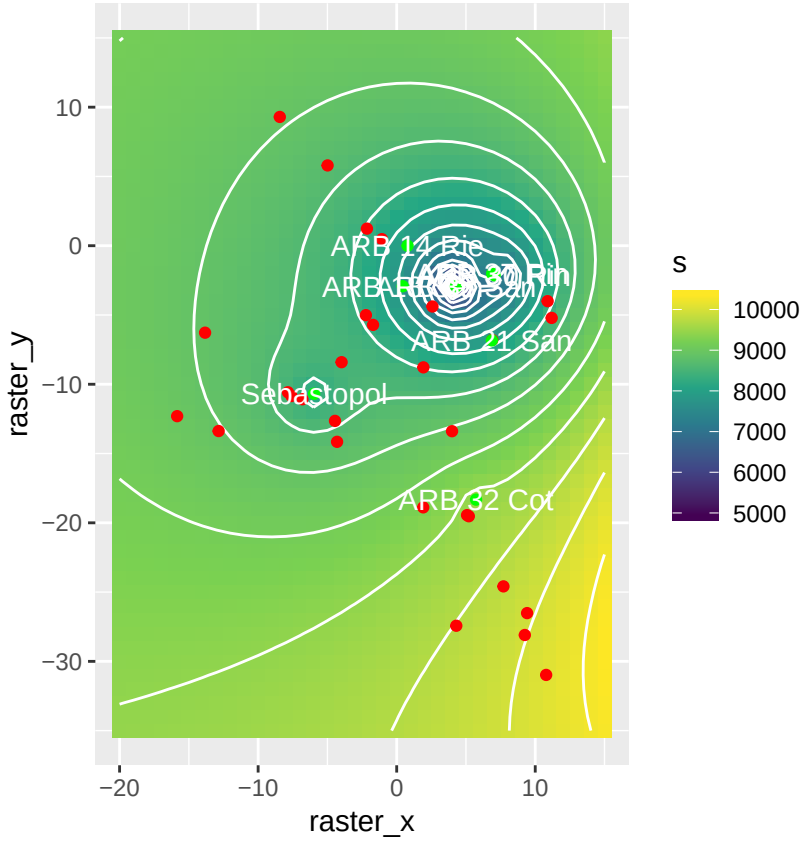
```
myIDW <- function(data = pm25, start_time = "2017-10-08", end_time = "2017-10-20",
                  xmin = -110, xmax = 110, xinc = 1, ymin = -110, ymax = 110, yinc = 1,
                  idp = 2, nmax = Inf, mdist = Inf) {

  stack <- list()

  j <- 0

  grid <- expand.grid(x = seq(xmin, xmax, xinc),
                      y = seq(ymin, ymax, yinc))

  coordinates(grid) <- ~x+y

  gridded(grid) <- TRUE

  for (i in seq.POSIXt(as.POSIXct(start_time), as.POSIXct(end_time), "hour")) {

    timepoint <- data %>%
      filter(datetime==i)

    coordinates(timepoint) <- ~easting_km+northing_km

    j <- j+1
```

```
    stack[[j]] <- as_tibble(idw(value~1, locations = timepoint, newdata = grid, idp = idp, nmax = nmax,
    stack[[j]][4] <- as.POSIXct(start_time) + 3600*(j-1)
    colnames(stack[[j]]) <- c("raster_x", "raster_y", "value", "datetime")

  }

  return(stack)

}
```

With all sensors

```
invisible(krige <- myIDW(xmin = -20, xmax = 15, xinc = 1, ymin = -35, ymax = 15, yinc = 1, idp = 1))

krige <- do.call(rbind, krige)

cumul <- krige %>%
  group_by(raster_x, raster_y) %>%
  summarize(s = sum(value))
```

```
cumul %>%
  ggplot(aes(raster_x, raster_y)) +
  scale_fill_viridis_c() +
  geom_raster(aes(fill = s)) +
  geom_contour(aes(z = s), bins = 20, color = "white") +
  geom_point(data = farms, aes(farms$easting_km, farms$northing_km), color = "red") +
  geom_point(data = distinct(pm25,lat,long,.keep_all=TRUE), aes(easting_km, northing_km), color = "gree
  geom_text(data = distinct(pm25,lat,long,.keep_all=TRUE), aes(easting_km, northing_km, label = key), c
  coord_fixed(xlim=c(-20,15),ylim=c(-35,15))
```




```
farms <- farms %>%
  mutate(raster_x = round(easting_km), raster_y = round(northing_km)) %>%
  left_join(rename(cumul, "allpm25" = s), c("raster_x", "raster_y"))
```