PWFSLSmoke Version 1.0

Helen Miller

2019-01-31

Mazama Science has released version 1.0 of the PWFSLSmoke R package for working with smoke monitoring data. The initial version was released last year, along with an accompanying blog post. In this post, we will discuss the purpose and uses of the R package and outline some new functionality in the new version.

The PWFSLSmoke package provides the tools to see how smoke affects communities across the country through analyzing and visualizing PM2.5 data. Its capabilities include:

- providing a versatile data model for dealing with PM2.5 data with the ws_monitor object
- loading real-time and archival raw or pre-processed PM2.5 data from permanent and temporary monitors
- quality-control options for vetting raw data
- mapping and plotting functions for visualizing long and short-term data
- algorithms for calculating NowCast values, rolling means, and aggregation
- functionality for manipulating and analyzing PM2.5 data

Why study PM2.5 data?

Mazama Science created the PWFSLSmoke package for the AirFire team at the USFS Pacific Wildland Fire Sciences Lab (PWFSL) as a tool to analyze and visualize data from PM2.5 monitoring stations all over the country. PM2.5 refers to particulate matter under 2.5 micrometers in diameter, and can come from many different sources, like car exhaust, power plants, or agricultural burning. Breathing air with high PM2.5 levels is linked to all sorts of cardiovascular diseases and can worsten or trigger conditions like asthma and other chronic respiratory problems. For many communities outside of large metro areas, the main source of PM2.5 is wildfire smoke, and PM2.5 levels regularly reach hazardous levels during wildfire season. 2017 was a particularly bad smoke year across the Pacific Northwest, and even large cities like Seattle, Portland, and San Fransisco felt the effects of wildfire smoke during the height of the fire season.

There are three main organizations that aggregate smoke data in the United States: AirNow, WRCC, and AIRSIS. PWFSLSmoke includes capabilities for ingesting, parsing, and quality-controlling raw data or loading RData files of pre-processed real-time and archival data from these sources.

Napa Valley Fires

The 2017 wildfire season in the US was one of the most destructive ever. California was hit particularly hard. Unless you've been living under a rock during the past few months, you have heard about the fires that tore through Napa wine country in October. They were incredibly destructive, destroying thousands of homes, businesses, wineries and vineyards, and taking close to 30 lives. Clouds of smoke choked communities miles away from the direct path of the fires, with concentrations climbing to unhealthy to hazardous levels.

We can see how smoke affected communities near the fires by using the PWFSLSmoke package to explore PM2.5 data from October of 2017. In the interest of readability and succintness, code for generating plots is excluded from the blog post, but can be found on GitHub.

Loading the data

Archival data can easily be loaded using `airnow_load()`. We can select monitors that are within 100 kilometers of the Tubbs fire, the largest of the Napa Valley fires, with `monitor_subset()` and `monitor_subsetByDistance()`.

Since we are looking at smoke from specific fires, let's load some fire data. Cal Fire has open archival fire data that we can use. There were three large fires in the Napa Valley region which all started on October 8th. Let's take a look at these fires in particular and data from monitors within 100km of the largest of the three.

```
suppressPackageStartupMessages(library(PWFSLSmoke))

# Fires started on Oct 9
fireDF <- data.frame(name = c("Tubbs", "Atlas", "Sulpher"),
                     startdate = parseDatetime(c(20171009, 20171009, 20171009)),
                     enddate = parseDatetime(c(20171031, 20171028, 20171026)),
                     longitude = c(-122.63, -122.24, -122.65),
                     latitude = c(38.61, 38.39, 39.01),
                     stringsAsFactors = FALSE)


CAFires <- airnow_load(year = 2017, month = 10) %>%
  monitor_subset(stateCode = "CA") %>%
  monitor_subsetByDistance(fireDF$longitude[1], fireDF$latitude[1], radius = 100)

str(CAFires, list.len = 4, width = 100)
```

```
## List of 3
##  $ meta    :'data.frame':    18 obs. of  19 variables:
##   ..$ monitorID           : chr [1:18] "069999917_01" "060130002_01" "060110007_01" "060750005_01"
##   ..$ longitude           : num [1:18] -123 -122 -122 -122 -122 ...
##   ..$ latitude            : num [1:18] 38.8 37.9 39 37.8 39.2 ...
##   ..$ elevation           : num [1:18] NA NA 244 2 NA 1 3 NA 6 20 ...
##   .. [list output truncated]
##  $ data    :'data.frame':    8760 obs. of  19 variables:
##   ..$ datetime    : POSIXct[1:8760], format: "2017-01-01 00:00:00" "2017-01-01 01:00:00" "2017-01-01
##   ..$ 069999917_01: num [1:8760] NA NA NA NA NA NA NA NA NA NA ...
##   ..$ 060130002_01: num [1:8760] 8 10 18 17 13 23 12 11 13 11 ...
##   ..$ 060110007_01: num [1:8760] 10.5 3.4 NA 34.5 1.9 3.5 3.9 0.1 0.7 1.2 ...
##   .. [list output truncated]
##  $ distance: Named num [1:18] 25.8 91.5 54 96 84.6 ...
##   ..- attr(*, "names")= chr [1:18] "069999917_01" "060130002_01" "060110007_01" "060750005_01" ...
##  - attr(*, "class")= chr [1:2] "ws_monitor" "list"
```

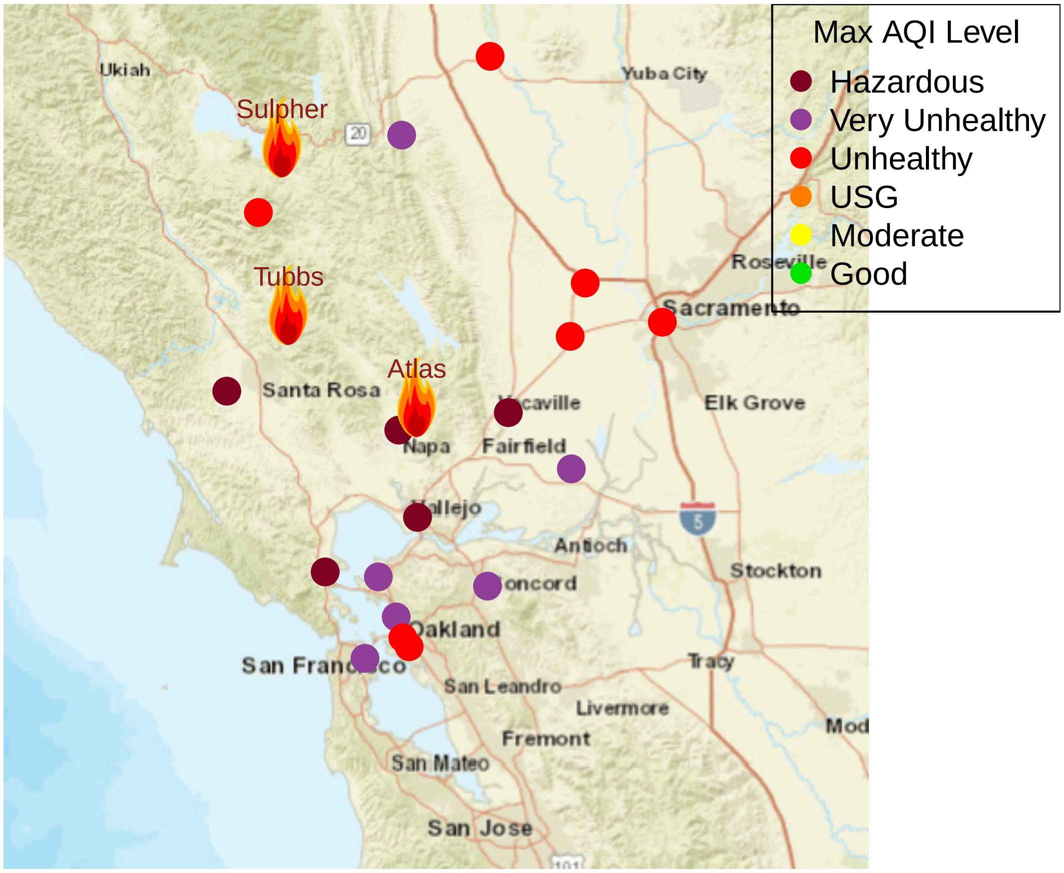
Now that we've got the data loaded into the environment, let's delve into PWFSLSmoke's plotting capabilities to see what it can tell us.

Mapping

A good place to start is by mapping monitor and fire locations. There are several different functions for mapping monitors. `monitor_leaflet()` will generate an interactive leaflet map that will be displayed in RStudio's 'Viewer' tab. There are two functions for creating static maps: `monitor_map()` will plot monitors over the outlines of states and counties, and `monitor_esriMap()` will plot monitors over a map image from ESRI.

Particulate concentrations can be classified by the Air Quality Index (AQI) that they fall into. AQI levels are defined by the EPA for regulating and warning people about air quality issues. The AQI cutoffs and the official colors associated with them are built into the package and used in several different plotting functions. All of the various monitor mapping functions color the monitor locations based on the AQI level of the maximum hourly PM2.5 value by default. There are some other built-in shapes which can be used in mapping, and added with the `addIcon()` function. For example, you could add the location of a real-life fire to a monitor map with a little picture of a fire as in the map below.

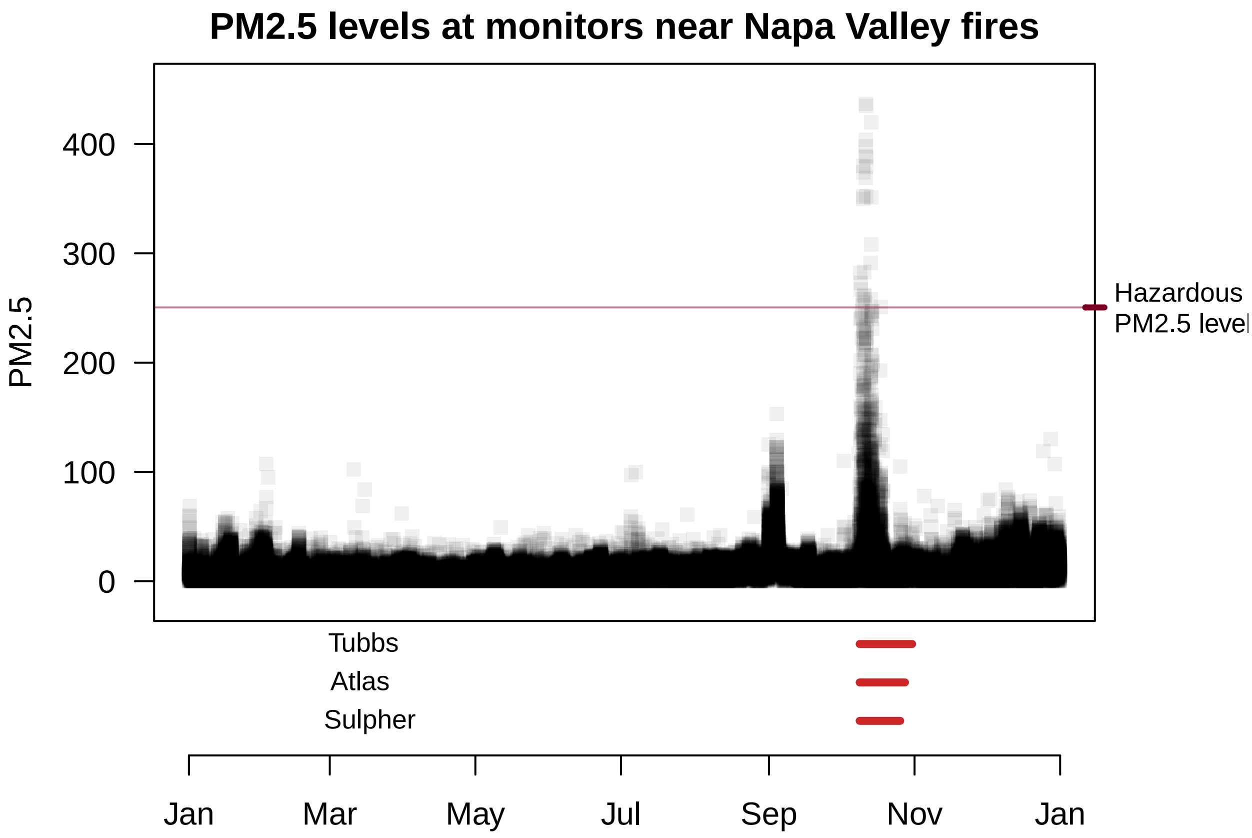
```
## INFO [2019-01-31 09:11:56] ESRI json URL: http://server.arcgisonline.com/arcgis/rest/services/World_S
## INFO [2019-01-31 09:11:56] ESRI json URL: http://server.arcgisonline.com/arcgis/rest/services/World_S
## INFO [2019-01-31 09:11:56] ESRI json URL: http://server.arcgisonline.com/arcgis/rest/services/World_S
## INFO [2019-01-31 09:11:56] ESRI json URL: http://server.arcgisonline.com/arcgis/rest/services/World_S
## INFO [2019-01-31 09:11:57] ESRI png URL: http://server.arcgisonline.com/arcgis/rest/services/World_St
## INFO [2019-01-31 09:11:57] ESRI png URL: http://server.arcgisonline.com/arcgis/rest/services/World_St
## INFO [2019-01-31 09:11:57] ESRI png URL: http://server.arcgisonline.com/arcgis/rest/services/World_St
## INFO [2019-01-31 09:11:57] ESRI png URL: http://server.arcgisonline.com/arcgis/rest/services/World_St
```



This map tells us that smoke reached hazardous levels in those communities closest to the fires. As far away as San Fransisco, smoke levels were very unhealthy. Of course, this does not tell us anything about when the smoke affected these communities or for how long.

Timeseries plots

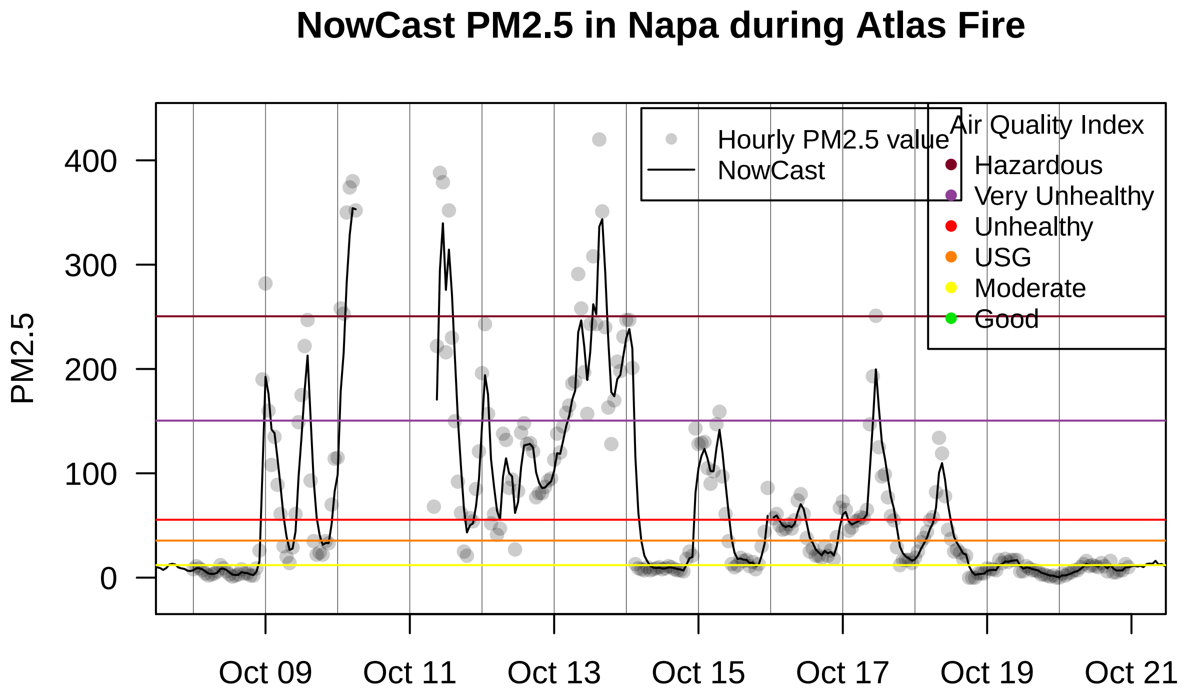
`monitor_timeseriesPlot()` is designed to plot timeseries data or visualizing data over time. It has a 'style' argument, with a couple of built-in plotting styles for telling different kinds of stories. The plot below uses the 'gnats' style. Red bars under the plot represent the duration of different fires.



This plot shows that PM2.5 levels were relatively constant around a baseline until shortly after the three fires started on October 9. The fires ignited and quickly built to sizes large enough to send thick smoke to all neighboring communities. The first couple of days were smokiest. Some monitors recoreded normal levels again around October 12 while many were still engulfed in smoke. After a brief respite, the baseline started creeping up again around October 16. All monitors returned to baseline levels around October 18, and stayed there for the rest of the month.

This gives us an idea of what air quality was like in the general region surrounding the fires. However, a curious observer of wildfire smoke might like to know how smoke affected a particular community. The city of Napa was directly hit by the Atlas fire, so let's take a look at smoke levels in Napa to see if it gives us any insight into the effects of the fires there.

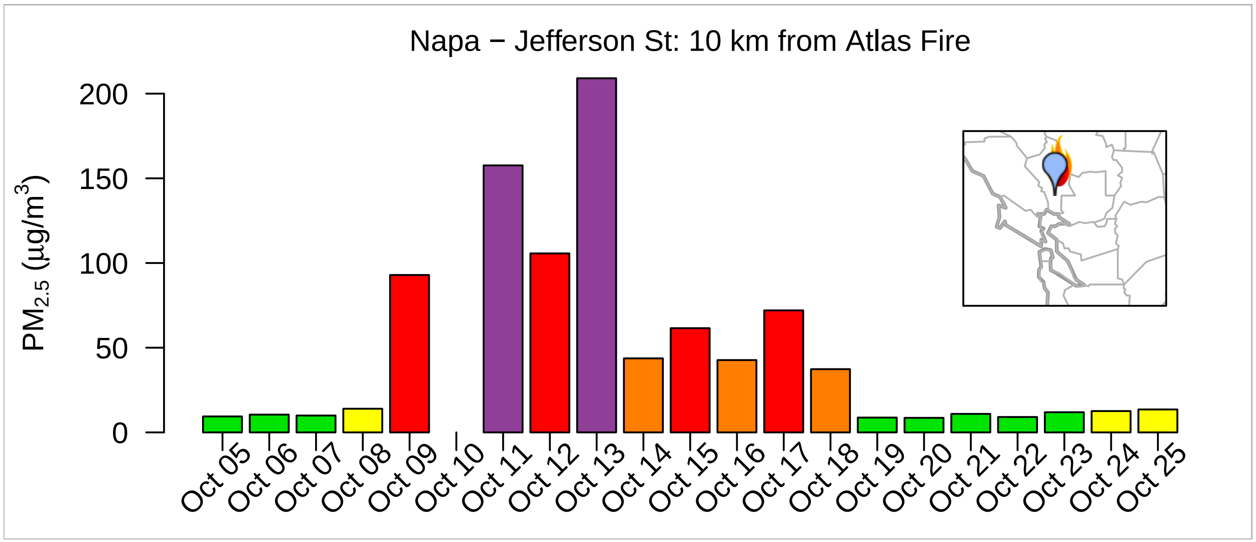
During this period, violent winds and flames meant that smoke levels could be jump wildly from hour to hour. One way to smooth out those changes a bit is by using NowCast values. NowCast is used for smoothing data and can be used to estimate values for missing data. The `monitor_nowcast()` function will calculate hourly NowCast values for a `ws_monitor` object. Using `monitor_timeseriesPlot()` to plot hourly values, and `monitor_nowcast()` to calculate and plot NowCast values on top of them, we can get a pretty good idea about what happened in Napa while the wildfires raged nearby.



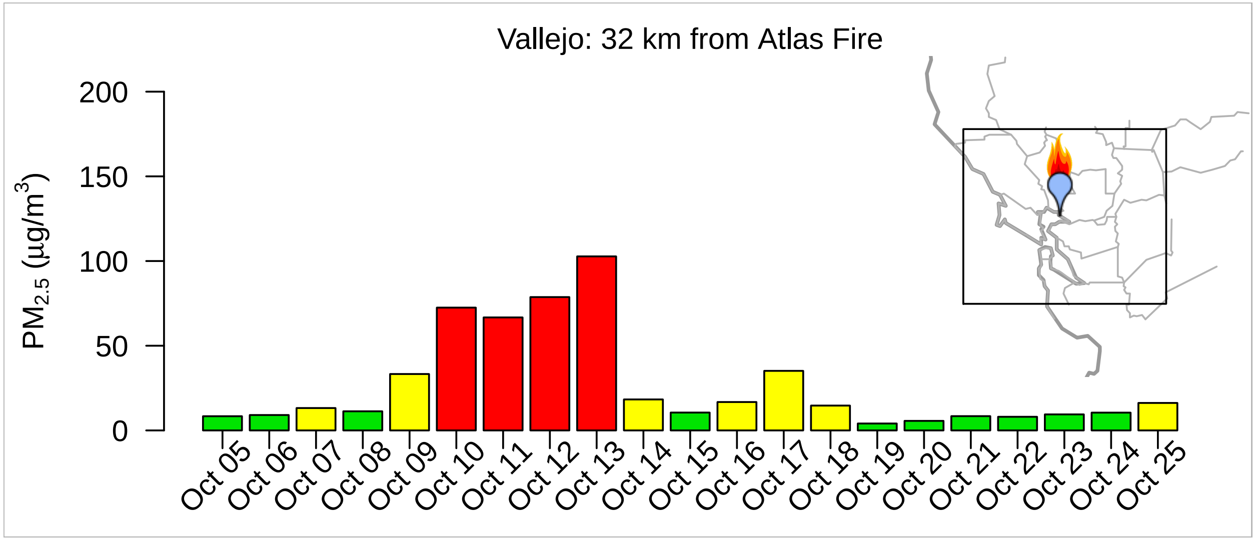
According to the data, PM2.5 levels were healthy up until close to midnight the night of October 8, when the Atlas Fire ignited and quickly exploded into a raging blaze only kilometers away. Smoke levels swung between moderate and very unhealthy throughout October 9, perhaps corresponding to changes in wind and weather. There is over a full day of missing values between October 10th and 11th, suggesting that the fire became so intense that the monitor was unable to record smoke values. It comes back online during the 11th, recording dangerously high PM2.5 values for several days, indicating that clouds of smoke kept billowing into the town before finally easing off around October 19th.

Plotting aggregated data

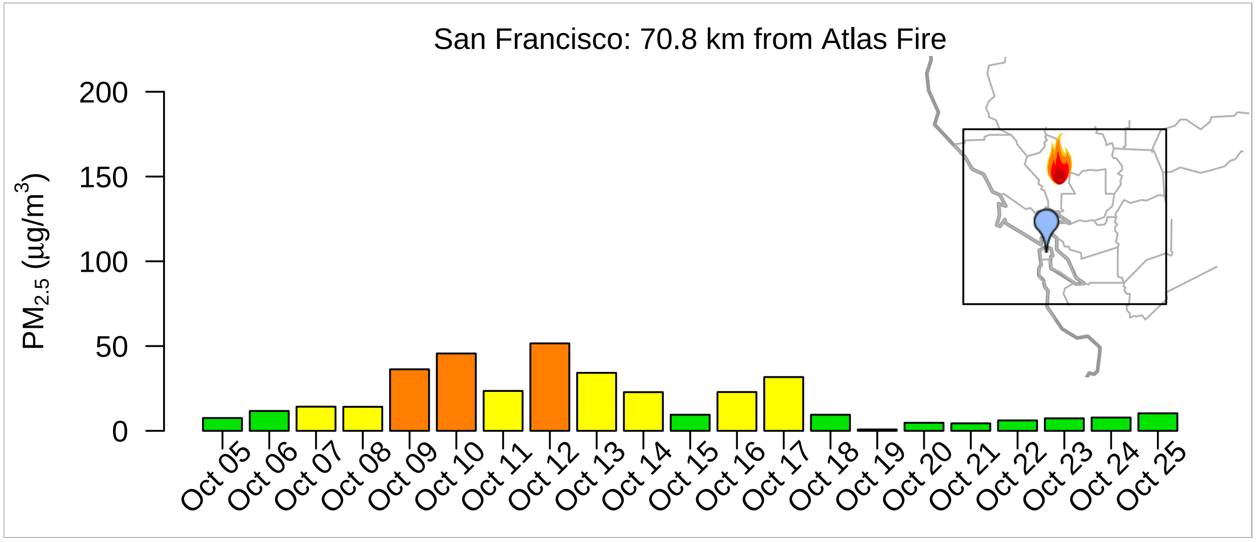
This gives us a pretty good idea about how smoke affected a community very close to a fire on a pretty detailed level. While examining a specific subset of the data like this might give us some important insights, viewing aggregated data can betray different stories. Let's say we want to get an idea about how far smoke from these fires traveled and how communities farther from the fires experienced it. One option is to look at how smoke levels differed in locations at varying distances from the fires. The plot below does this, using `monitor_dailyBarplot()` to plot the mean PM2.5 value for each day at Napa, Vallejo, and San Fransisco, with the distance from the Atlas Fire calculated using `monitor_distance()`.



Napa − Jefferson St: 10 km from Atlas Fire



Vallejo: 32 km from Atlas Fire



San Francisco: 70.8 km from Atlas Fire

The shape of the data for the three different locations is pretty similar, with overall PM2.5 levels decreasing as the distance increases. Unfortunately, we are missing a day's worth of data at Napa. However, the information from other monitors might help us guess what it was. At both Vallejo and San Fransisco, there was an increase in smoke from October 9 to 10, and a decrease from October 10 to 11. If smoke in Napa followed the same pattern, which would make sense if the smoke is coming from the same source in all three cities, we could speculate that smoke levels on October 10 would probably be between the values for October 11 and October 13.