Analyzing AQI Data in Los Angeles

Summary

In this analysis, I pulled a series of monthly AQI readings in the Los Angeles area from NASA's EarthData website, then performed a time series analysis on the data to forecast future readings. The most promising results come from the model using Facebook's Prophet library, while a SARIMA model is also considered but results in a higher Mean Absolute Error.

Background

NASA publishes historical records of aerosol readings from the Goddard Earth Observing System on their EarthData site; this data is freely accessible after creating a login account. Among the public data sources are monthly recordings of air particulate matter, used in calculating well-established metrics such as the EPA's Air Quality Index (AQI), the most well-known among these being the PM2.5, which is the amount of particulate matter in the air 2.5 micrometers or smaller. This particulate matter can have negative effects on public health, so it is useful to understand the trends in concentration levels over time, as well as forecasting what future levels will look like.
In this report, I will run a Time Series Analysis on the monthly levels of PM2.5 by examining the historical trends in this data and trying out different forecasting models to see how well they perform.

Approach

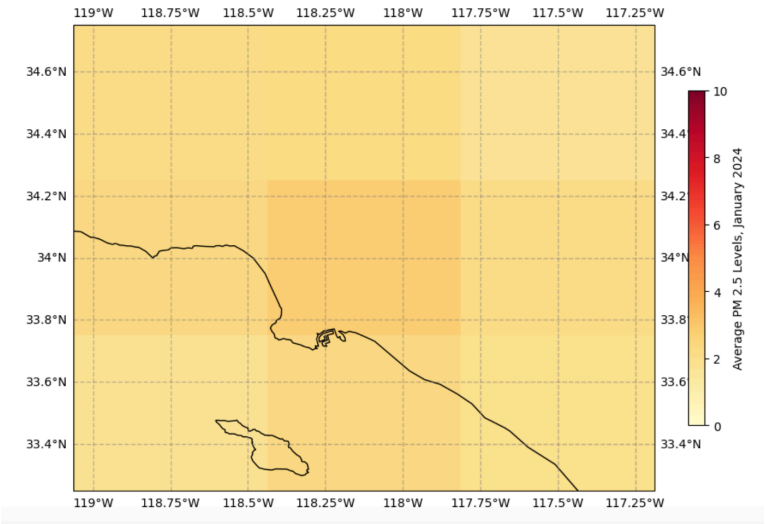
After pulling data from the past 20 years and performing some cleaning and processing steps, I will proceed to explore the shape and structure of the data to understand what I'm working with, followed by some attempts to fit time series models that I use to forecast future data.
The code for completing this analysis takes place in a series of 3 Jupyter notebooks:

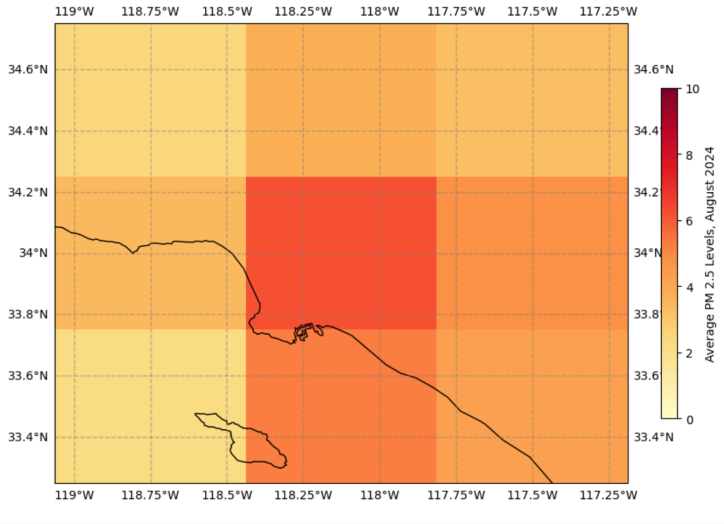
- api_request.ipynb: code for calling the EarthData api to retrieve the data files
- EDA.ipynb: notebook that runs the api_request code to download and save 20 years worth of data then runs Exploratory Data Analysis to get a feel of the data
- AQ Time Series: notebook for building and running the time series models based on the data saved from EDA.ipynb

Exploration

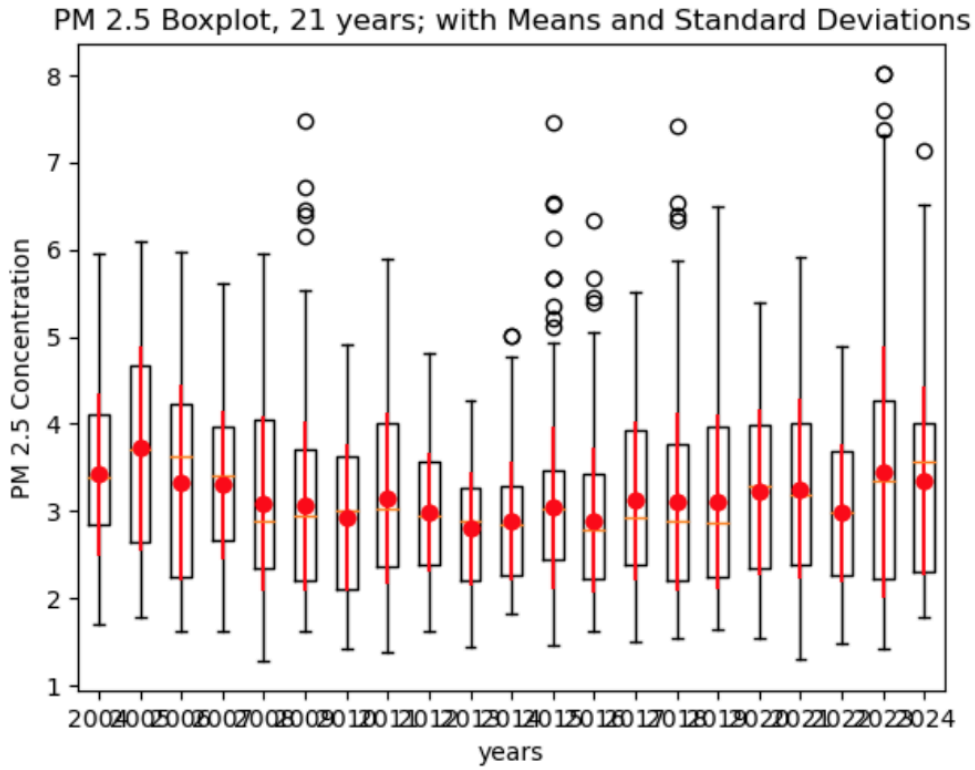
NASA has provided detailed instructions and examples for accessing data through their APIs, so I modified their initial notebook for my purposes (see the api_request notebook) to be able to access my desired dataset, the tavg1_2d_aer_Nx, using my login credentials. With this framework, I proceeded to pull the 20 years' worth of data and start the exploratory data analysis (performed in the EDA notebook).
As a starting point and to give context for the region I am examining, I plotted heat maps of the average monthly PM2.5 levels in January 2024, shown on the left, and August 2024, shown on

the right. Already we can see a noticeable increase in PM2.5 during the summer month compared to the winter.

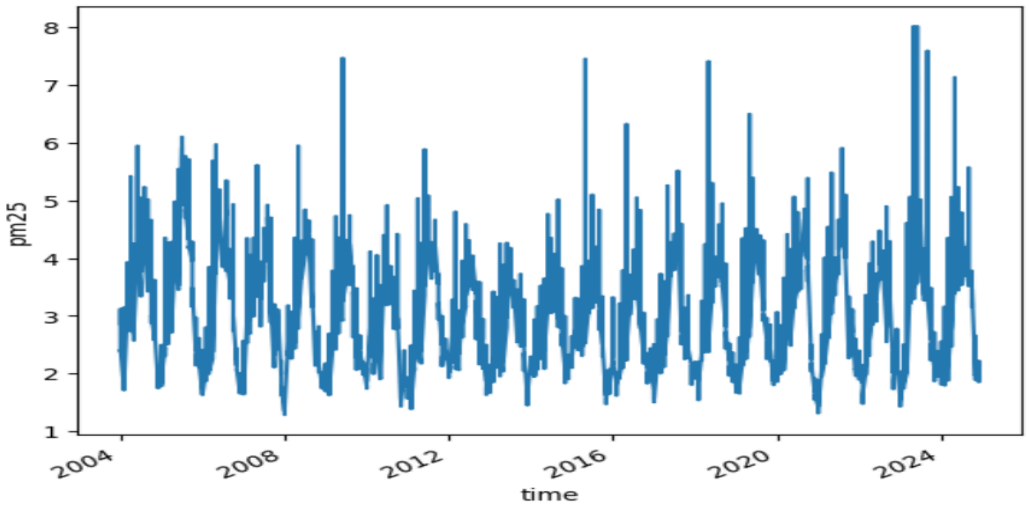




Next, I looked at the boxplot of the data readings on a per-year basis. The figure shows that the mean of each year has not changed considerably, though there are some years that have a lot more variance than others, such as 2014 and 2017



Finally, plotting the full time series shows us regular spikes and troughs that occur throughout each year.

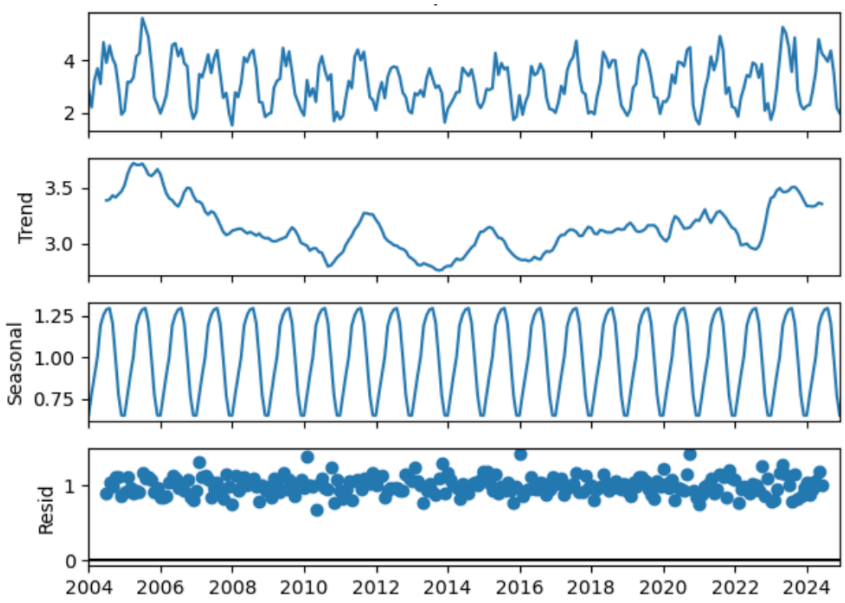


Modeling and Analysis

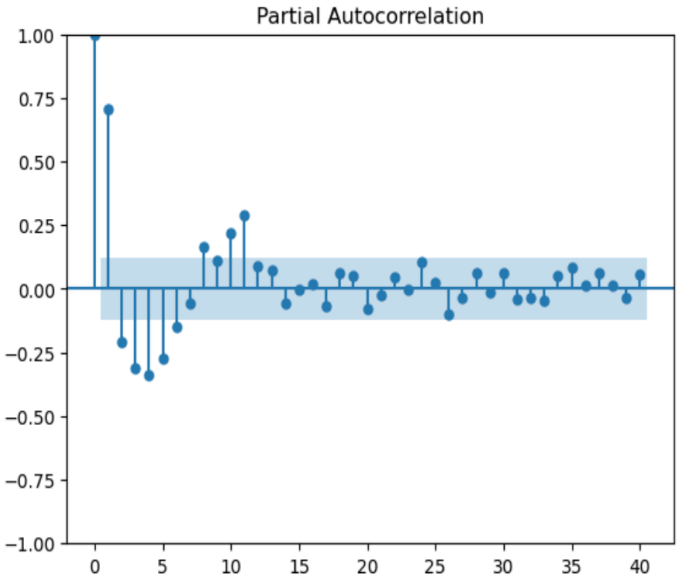
With the exploration completed, I was able to move on to building a model that would capture the signal in the data and forecast future PM25 results. This was performed in the AQ Time Series notebook.

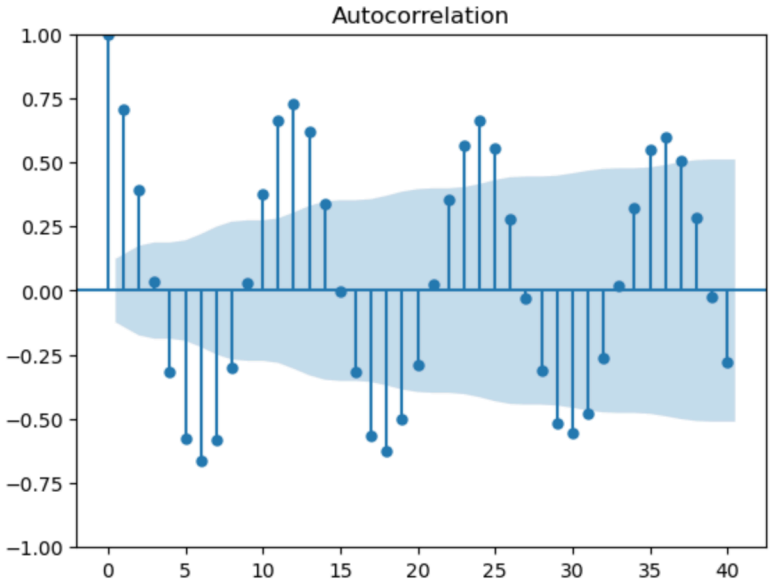
ARIMA approach

In approaching a model for this time series data, I first took a look at the ETS decomposition to understand the components of the series. Doing so shows very clear seasonal behavior. I will want to account for this in my modeling, particularly when I use ARIMA methods.
Examining the trend section, I notice that there may be a slight positive trend, although this may largely just be a result of an increase at the end of the series.



Taking a look at the ACF and PACF further confirms the seasonality, as there are spikes at the lags of 12 and 24 and troughs at the lags of 6 and 18, aligning with a yearly seasonality.





With all this in mind, I first split my data into a train/test dataset with the test containing the most recent 12 months of my data, then I used python's statsmodels libraries to perform a grid search in fitting various SARIMA (Seasonal ARIMA) models on the train dataset, relying on the AIC metric to determine the model with the best fit. I will use the test dataset below to see how well the models forecast future readings.

The model selected was a $(2,0,2)(0,0,0)_{12}$ SARIMA model, meaning it makes use of AR and MA models with lags to the second degree, 0 orders of differencing, and a seasonality of 12. The summary is shown below

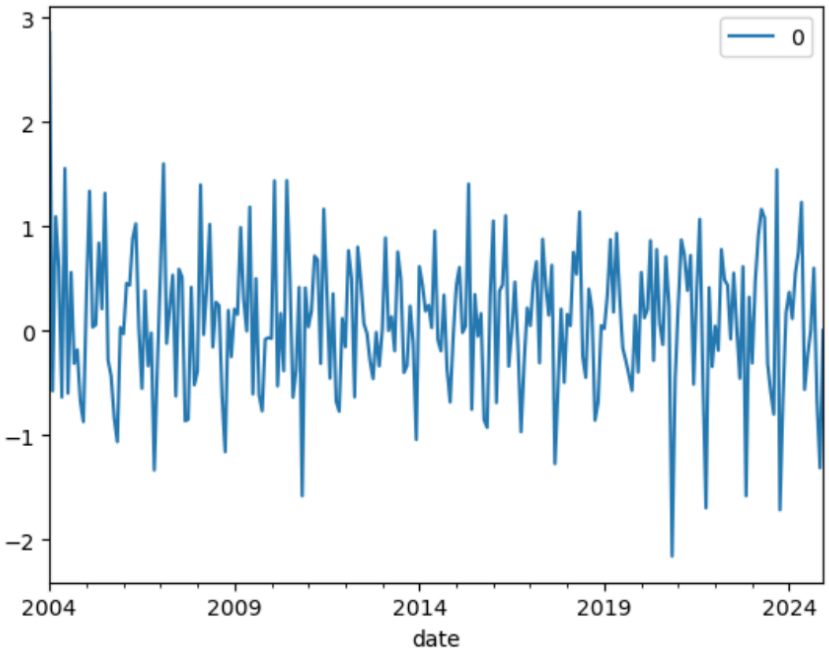
SARIMAX Results

Dep. Variable:	pm25	No. Observations:	252
Model:	SARIMAX(2, 0, 2)	Log Likelihood	−253.835
Date:	Sun, 02 Feb 2025	AIC	517.670
Time:	10:28:44	BIC	535.317
Sample:	01-01-2004	HQIC	524.771
	- 12-01-2024		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	1.2344	0.409	3.020	0.003	0.433	2.035
ar.L2	−0.2595	0.398	−0.653	0.514	−1.039	0.520
ma.L1	−0.2332	0.397	−0.587	0.557	−1.012	0.545
ma.L2	0.1618	0.077	2.091	0.037	0.010	0.313
sigma2	0.4334	0.038	11.432	0.000	0.359	0.508

Ljung-Box (L1) (Q):	0.02	Jarque-Bera (JB):	3.08
Prob(Q):	0.90	Prob(JB):	0.21
Heteroskedasticity (H):	0.98	Skew:	−0.24
Prob(H) (two-sided):	0.93	Kurtosis:	3.26

As a check on this, I plotted the residuals to make sure that there were no obvious signs of signal remaining.



Alternate Model

As an additional step, I tried out an alternate forecasting model, the [Prophet forecasting library], open-sourced by Facebook's Core Data Science Team, to see how it compares with my SARIMA model in forecasting on data out of our training sample.
After building this model, I forecasted the PM2.5 results for the year 2024 using both this model and the SARIMA model from earlier, then compared it against the ground truth results of my test dataset by calculating the Mean Absolute Error (MAE). The results were as follows:

```
10:31:57 - cmdstanpy - INFO - Chain [1] start processing
10:31:57 - cmdstanpy - INFO - Chain [1] done processing
Prophet MAE: 0.581
SARIMA MAE: 2.023
```

Based on these results, the model from Prophet actually performs better than my SARIMA model in terms of accuracy, so if we wanted to primarily focus on accuracy of prediction I would use this model instead of the SARIMA.

Conclusion and Next Steps

The model built using Facebook's Prophet library performed much better than our selected SARIMA model on forecasting future values of PM2.5: in terms of MAE, Prophet resulted in a value of 0.581 vs 2.023 for the SARIMA.

For some next steps on ways to improve our model, we could try expanding our data horizon or increasing the frequency from monthly to daily. We could also expand our grid search to allow for a SARIMA model with more parameters. Furthermore, we could try out new forecasting models such as IBM's TinyTimeMixers, which uses an architecture based on multi-layer perceptron modules. Ultimately, this would all depend on what our final goal is, and how much additional resources we wanted to spend on this endeavor.