AN ANALYSIS OF

AIR POLLUTION IN SEOUL

by Rohan Kalantri

THE DATA

Source - https://www.kaggle.com/bappekim/air-pollution-in-seoul
User - bappe

Files:

- *Measurement Items* - Describes the compounds being measured, their unit of measurement and severity range.
- *Station Info* - Describes the Station's address, its code and its coordinates
- *Measurement Info* - Describes the Instrument status at a given date and time and the average values captured
- *Measurement Summary* - Combines parameters from all of the above files into one single summary file

KEY VALUES

- *Shape of Dataset*- 647511 x 11
- *Measurement Dates* - 1 JAN 2017 - 31 DEC 2019
- *Measurement Frequency* - Once every hour - 24 measurements per parameter per day
- *Total Measurement Stations* - 25 - One per district in Seoul
- *Null Values* - Represented by -1

LIBRARIES & TOOLS

- Pandas - Cleaning, Querying, and manipulation of data
- Numpy - Value Manipulation
- Sklearn - Imputer
- Seaborn - Matrix
- Plotly - for plotting
- Matplotlib - for plotting
- Mapbox - base maps for geographic plotting

CLEANING

Split Measurement Date to Date and Time columns

```
# since the measurement date contains the date and time,
# we will split it into two different columns for more usability

summary[['date','time']] = summary['Measurement date'].str.split(expand=True)
```

CLEANING

Split the Address Column

```
# the address in the dataframe can be split to get the district names
# we get 5 different columns from the address after spliting it
summary[['number','area', 'district', 'city', 'country']] = summary['Address'].str.split(", ", expand=True)
```

CLEANING

Replace Null Values (-1) with the average values using *SimpleImputer*

```
# Null values (-1 in this dataset) are replaced with the avg. value of appropriate column

imputer = SimpleImputer(missing_values = -1, strategy = "mean")

imputer = imputer.fit(summary[["SO2", "NO2", "O3", "CO", "PM10", "PM2.5"]])

new = imputer.transform(summary[["SO2", "NO2", "O3", "CO", "PM10", "PM2.5"]])

new = pd.DataFrame(data = new)

new.columns = summary[["SO2", "NO2", "O3", "CO", "PM10", "PM2.5"]].columns
new.index = summary.index

summary['SO2'] = new['SO2']
summary['NO2'] = new['NO2']
summary['O3'] = new['O3']
summary['CO'] = new['CO']
summary['PM10'] = new['PM10']
summary['PM2.5'] = new['PM2.5']

summary[["SO2", "NO2", "O3", "CO", "PM10", "PM2.5"]].describe()
```

NEW FEATURES!

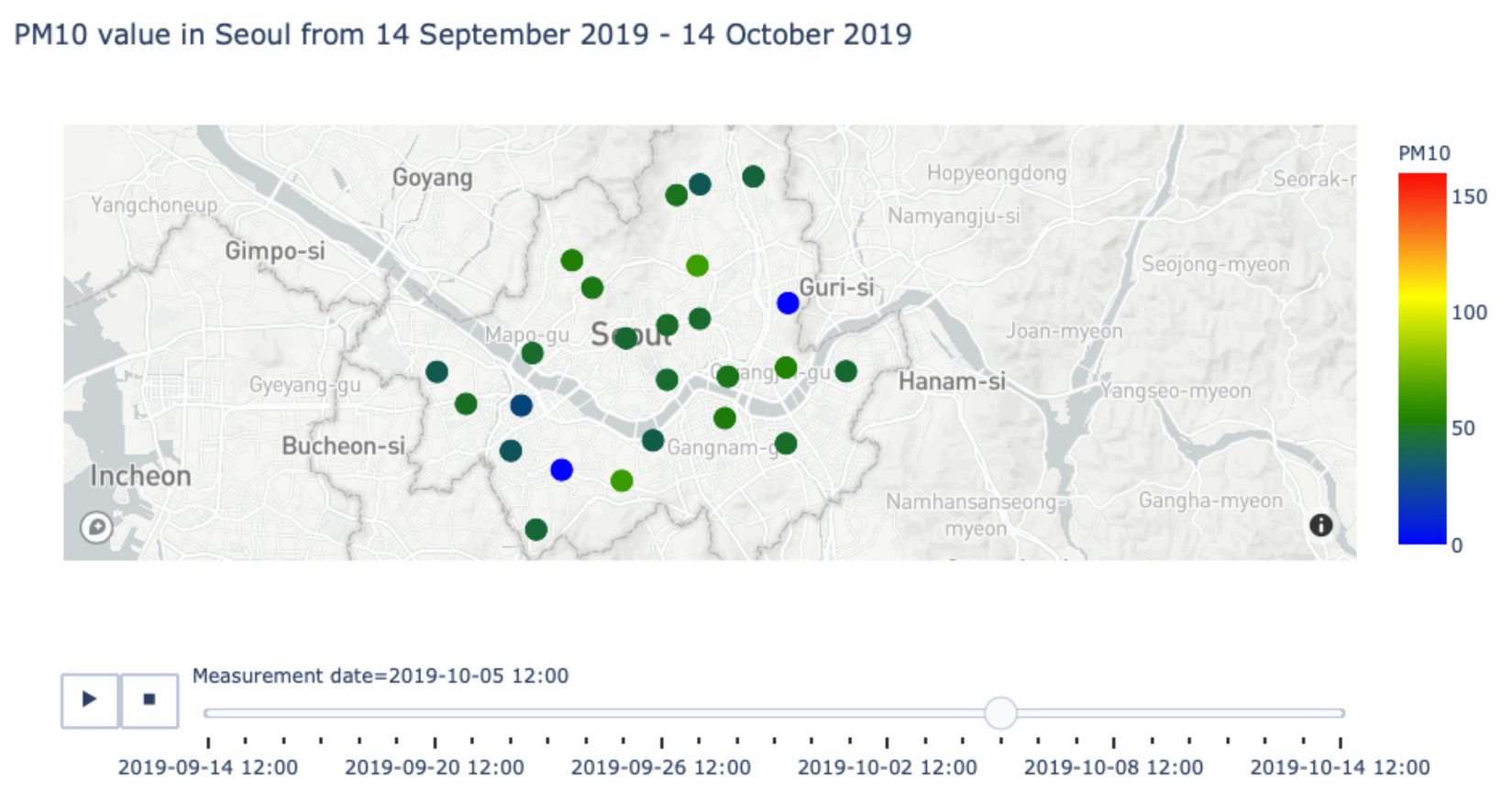
Add Columns to represent Severity of the compounds

```
summary['SO2 Severity'] = summary.apply(lambda row: severitySO2(row['SO2']), axis=1)

summary['NO2 Severity'] = summary.apply(lambda row: severityNO2(row['NO2']), axis=1)

summary['CO Severity'] = summary.apply(lambda row: severityCO(row['CO']), axis=1)

summary['O3 Severity'] = summary.apply(lambda row: severityO3(row['O3']), axis=1)

summary['PM10 Severity'] = summary.apply(lambda row: severityPM10(row['PM10']), axis=1)

summary['PM2.5 Severity'] = summary.apply(lambda row: severityPM25(row['PM2.5']), axis=1)
```

SCATTER PLOT

Seoul International Fireworks Festival took place from 1 - 5 Oct

Hypothesis - Increase in air pollution around this time



Click here to see the animation

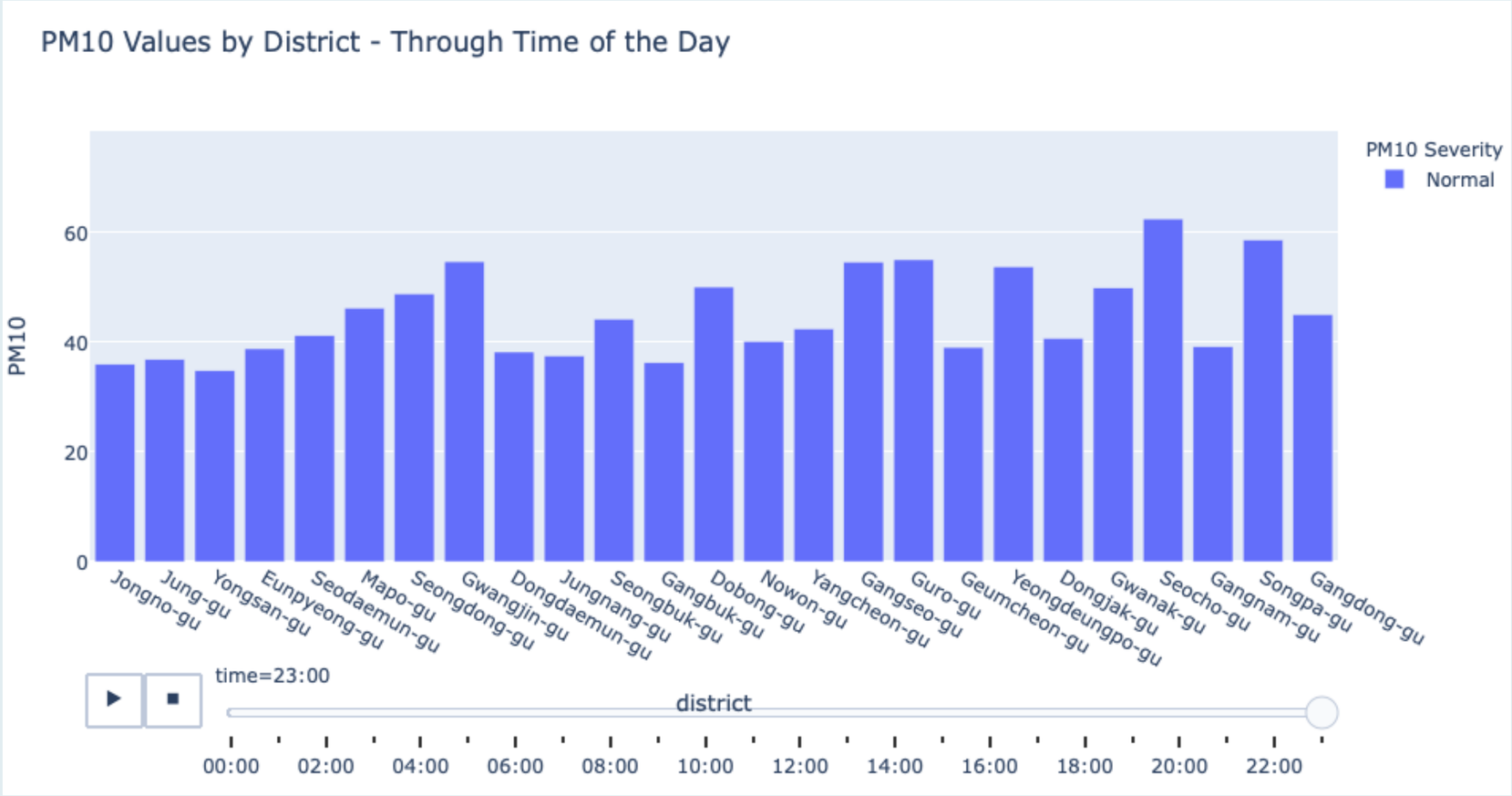
FILTER

Finding the average value of each parameter in every hour to understand how parameters change throughout the day

```
for st in summary['Station code'].unique():
    stdf = summary[summary["Station code"] == st]
    for time in stdf['time'].unique():
        timedf = stdf[stdf['time'] == time]
        so2 = np.average(timedf['SO2'])
        no2 = np.average(timedf['NO2'])
        o3 = np.average(timedf['O3'])
        co = np.average(timedf['CO'])
        pm10 = np.average(timedf['PM10'])
        pm25 = np.average(timedf['PM2.5'])
        district = timedf.iloc[0]['district']

        values_dict = {
            "time" : time,
            "station" : st,
            "district" : district,
            "SO2" : so2,
            "NO2" : no2,
            "O3" : o3,
            "CO" : co,
            "PM10" : pm10,
            "PM2.5" : pm25,
            "SO2 Severity": severitySO2(so2),
            "NO2 Severity": severityNO2(no2),
            "O3 Severity": severityO3(o3),
            "CO Severity": severityCO(co),
            "PM10 Severity": severityPM10(pm10),
            "PM2.5 Severity": severityPM25(pm25)
        }

        rows.append(values_dict)
```

BAR PLOT

Bar Plot of PM 10, segregated by districts, animated by hour of the day, coloured by severity



Click here to see the animation

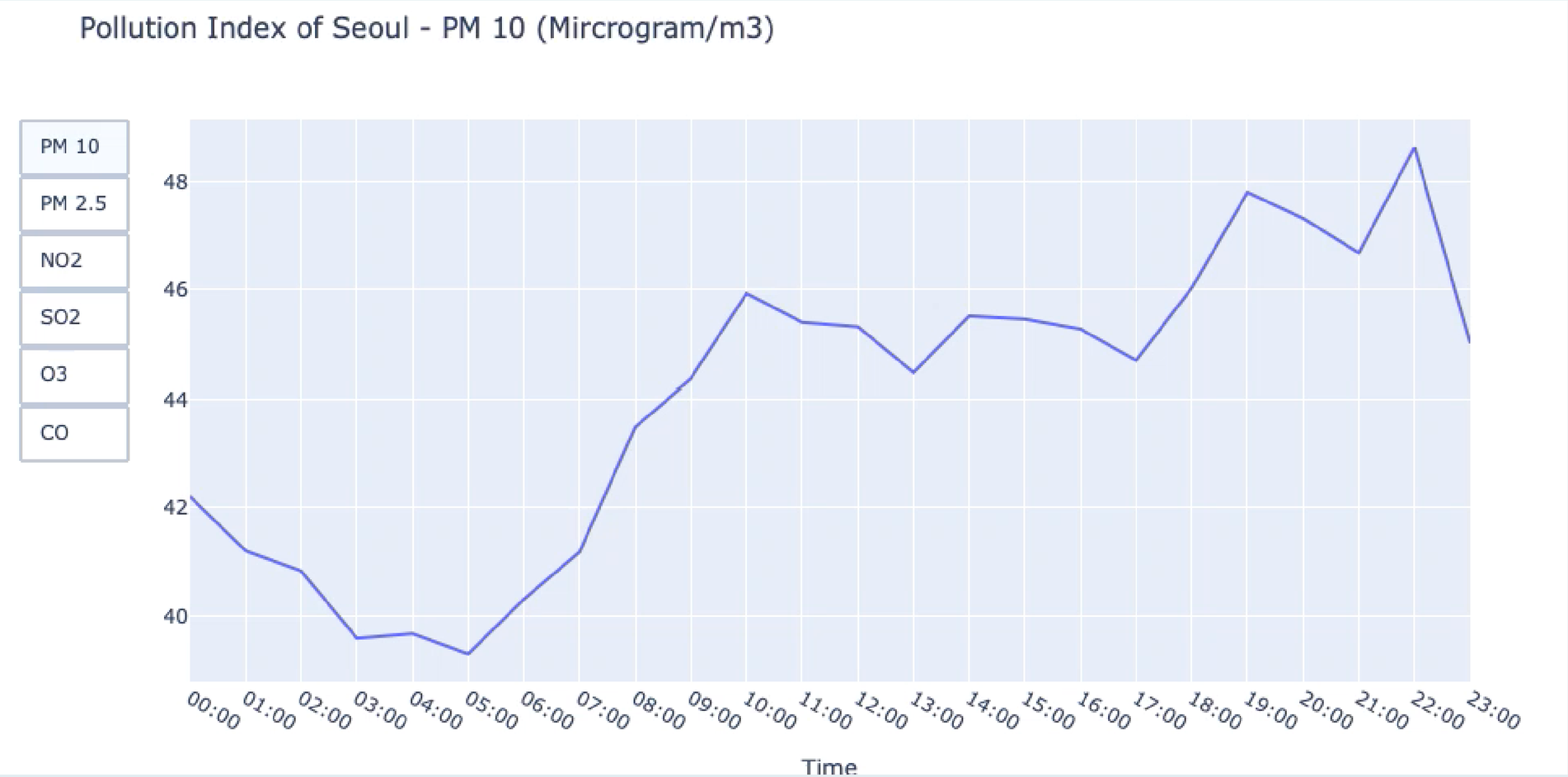
FILTER

Calculating average value of each parameter in each hour

```
for st in summary['Station code'].unique():
    stdf = summary[summary["Station code"] == st]
    for time in stdf['time'].unique():
        timedf = stdf[stdf['time'] == time]
        so2 = np.average(timedf['SO2'])
        no2 = np.average(timedf['NO2'])
        o3 = np.average(timedf['O3'])
        co = np.average(timedf['CO'])
        pm10 = np.average(timedf['PM10'])
        pm25 = np.average(timedf['PM2.5'])
        district = timedf.iloc[0]['district']

        values_dict = {
            "time" : time,
            "station" : st,
            "district" : district,
            "SO2" : so2,
            "NO2" : no2,
            "O3" : o3,
            "CO" : co,
            "PM10" : pm10,
            "PM2.5" : pm25,
            "SO2 Severity": severitySO2(so2),
            "NO2 Severity": severityNO2(no2),
            "O3 Severity": severityO3(o3),
            "CO Severity": severityCO(co),
            "PM10 Severity": severityPM10(pm10),
            "PM2.5 Severity": severityPM25(pm25)
        }

        rows.append(values_dict)
```

LINE PLOT

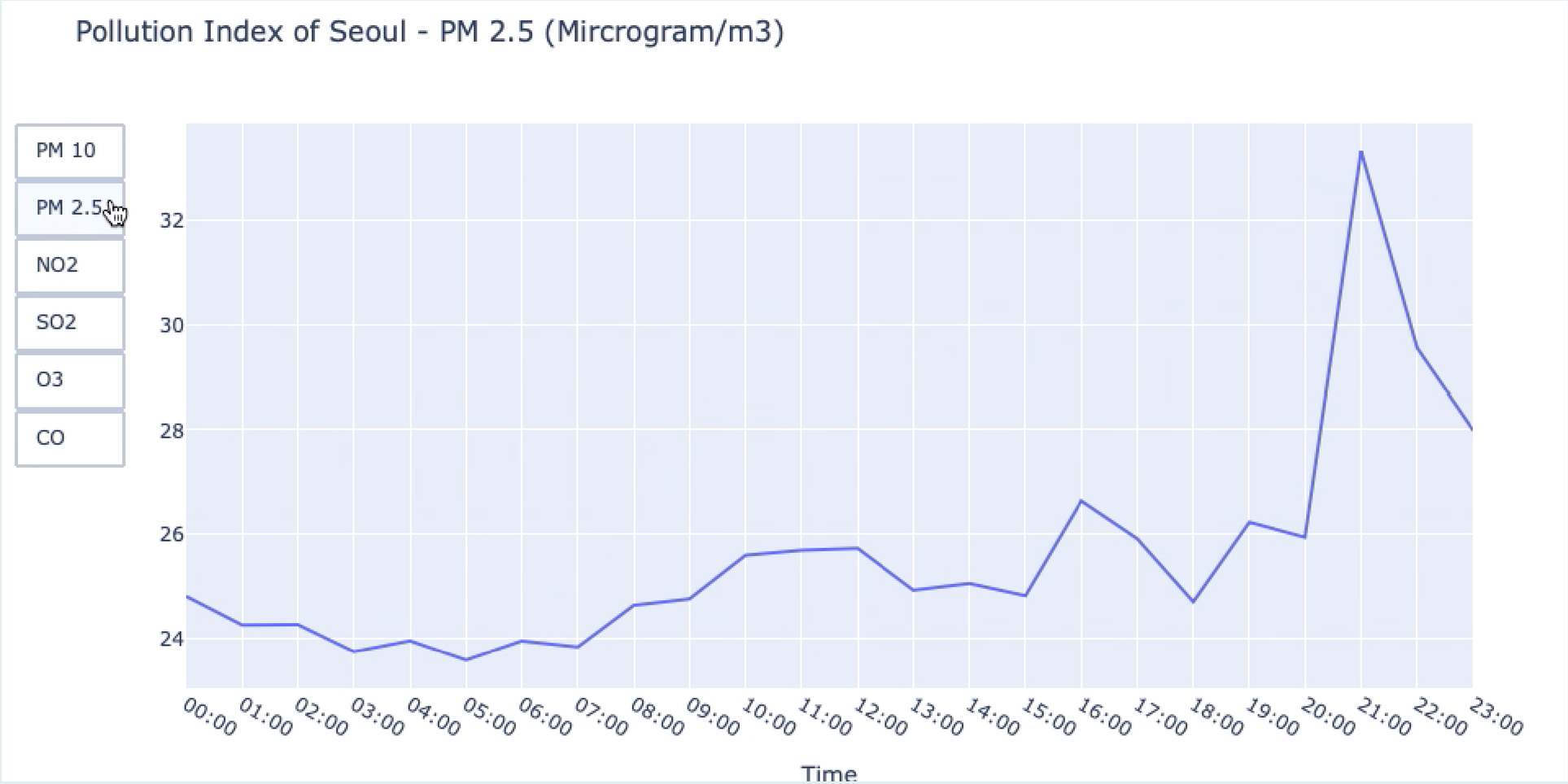
Creating a line plot of values of parameters (averaged per hour) - PM10



Click here to see the animation

LINE PLOT

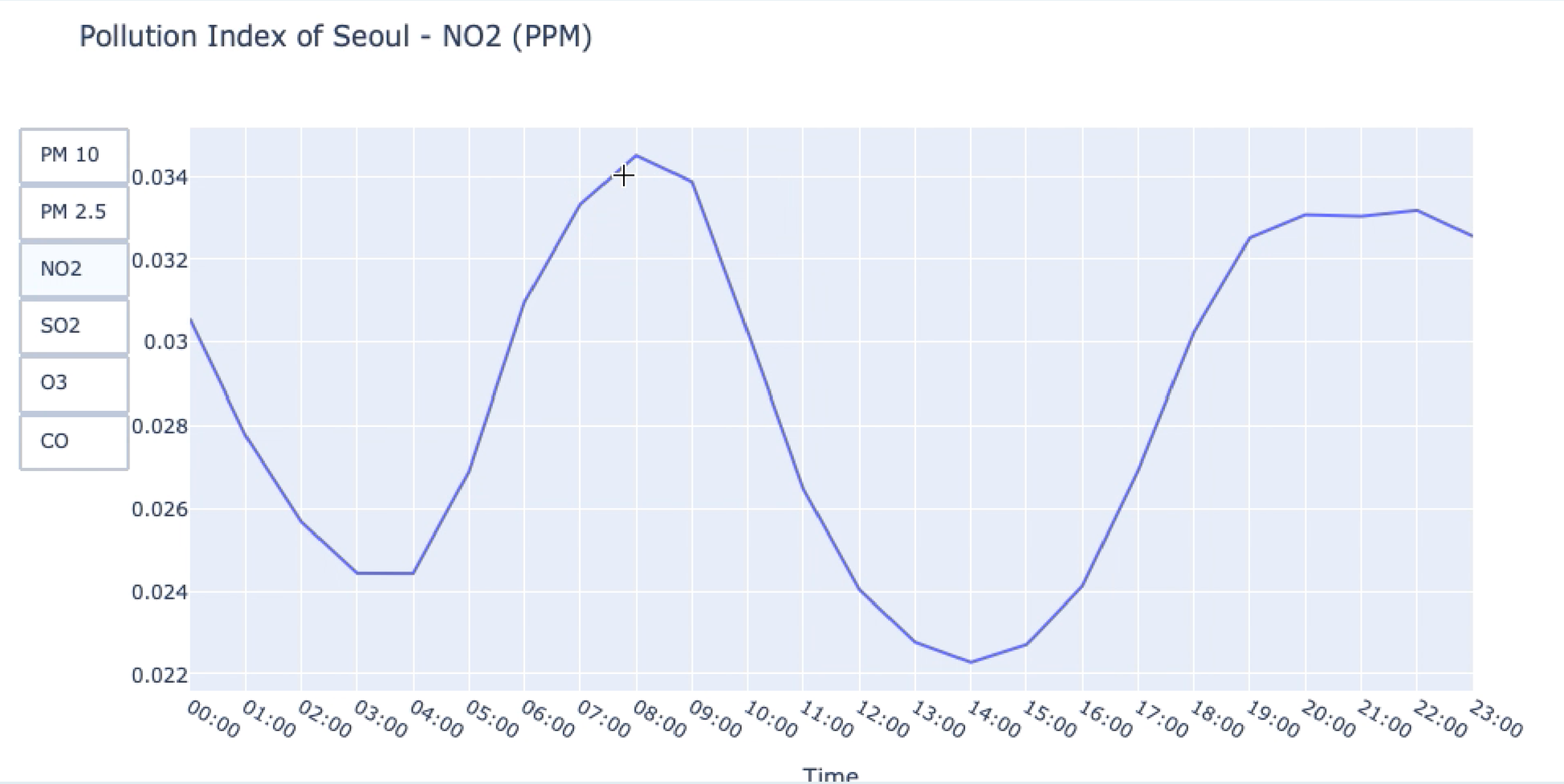
Creating a line plot of values of parameters (averaged per hour) - PM2.5



Click here to see the animation

LINE PLOT

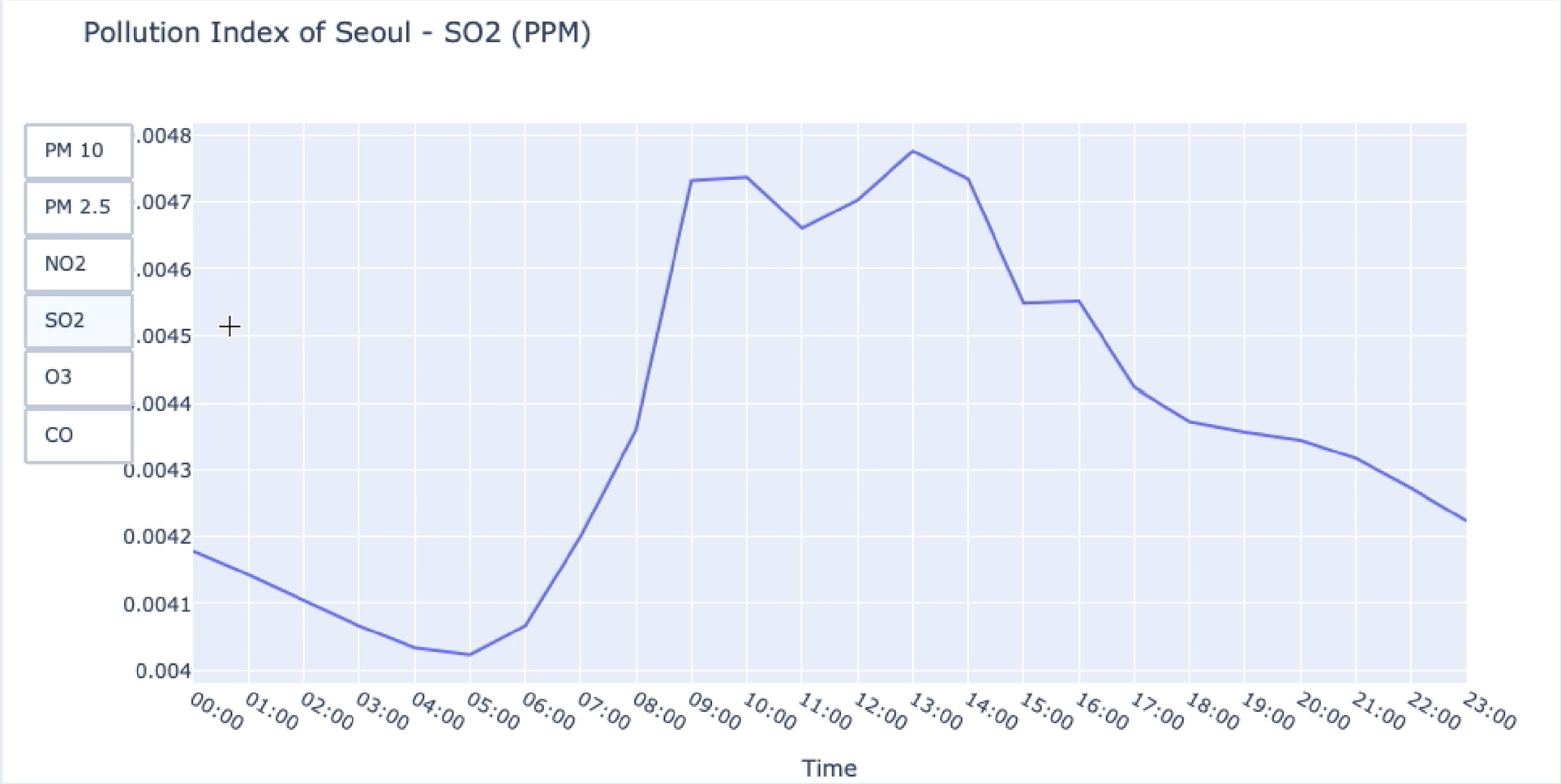
Creating a line plot of values of parameters (averaged per hour) - NO2



Click here to see the animation

LINE PLOT

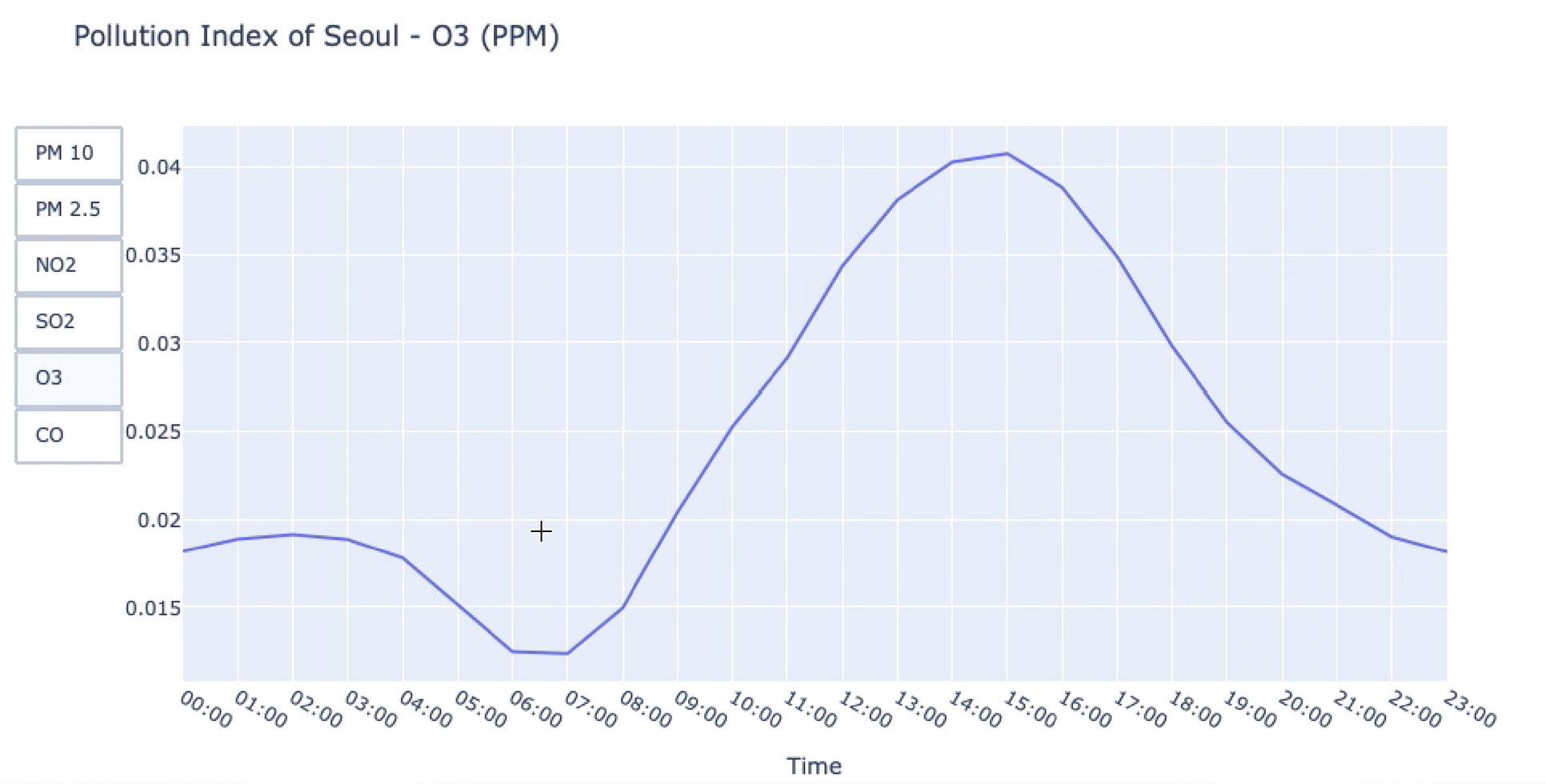
Creating a line plot of values of parameters (averaged per hour) - SO2



Click here to see the animation

LINE PLOT

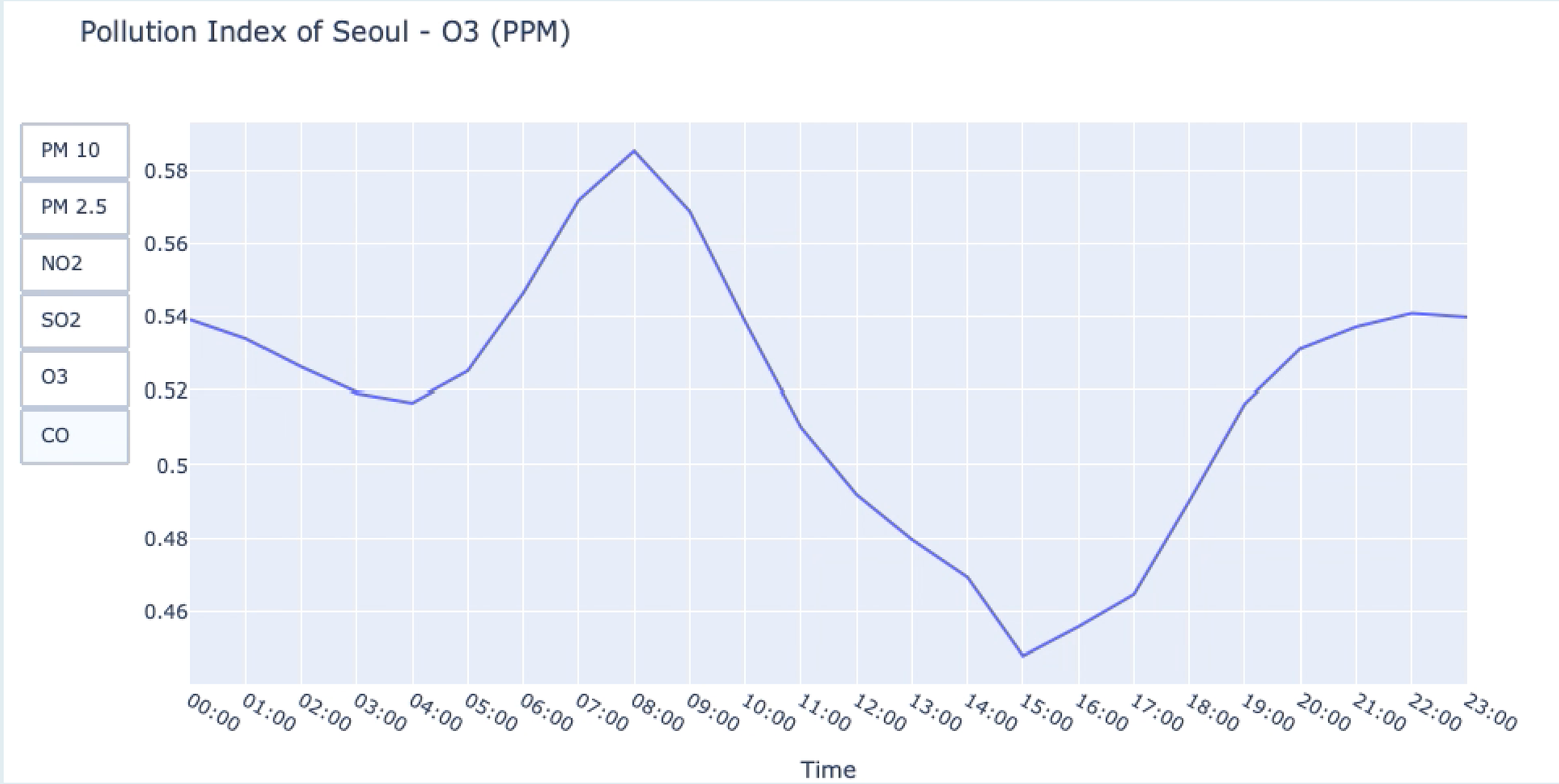
Creating a line plot of values of parameters (averaged per hour) - O3



Click here to see the animation

LINE PLOT

Creating a line plot of values of parameters (averaged per hour) - CO



Click here to see the animation

INFERENCES

Looking at the Scatter Plot,

- Since the plot was made keeping in mind the Seoul International Fireworks Festival, an increase in pollution by the end of the festival was expected.
- This hypothesis is proven to be true when the PM10 value goes up on the 5th of October
- Similarly, the values for other parameters too, goes up at this time.

Looking at the hourly average plots,

- NO2, SO2, O3 and CO have a sine wave like curve. A closer look will show us that just before and right after the working hours - these touch the crests.
- These are the times when the movement of traffic is high due to travel to and from work.
- PM10 and PM2.5 show an overall upward trend - until the end of the day when the value reduces again.
- Overall, the air pollution in Seoul is quite low. Each of the parameters are in the Good or Normal range.

FUTURE PROSPECTS

- It is important to take into consideration, the instrument status at the given date and time. This will help in dealing with noisy data appropriately.
- Trying to find out more events like the Fireworks festival to further support our hypothesis.
- Getting a better understanding of the parameters in greater depth and what factors affect their presence in the atmosphere.
- Comparing this data with data collected during the lockdown periods as a result of COVID19 pandemic.

THANK YOU

Made By:
Rohan Kalantri
Contact - kalantri.rohan@gmail.com